APPLE DOS
磁碟作业系统详论



康蜀嶼
蔡毓琛 編著

协群科技出版社

APPLE DOS
磁碟作業系統詳論

康蜀嶼
蔡毓琛　編著

協群科技出版社

APPLE DOS
磁碟作業系統詳論

編譯者： 康蜀嶼 · 蔡毓琛

出　版： 協群科技出版社

發　行： 協群科技出版社

香港中環卑利街684號二樓

印刷者： 廣源印務局

青山道875號工廠大厦

定價： H.K.$ 25.00

APPLE DOS
磁碟作業系統詳論

康蜀嶼
蔡毓琛 編著

協群科技出版社

APPLE DOS
磁碟作業系統詳論

編譯者：康司嶼 · [illegible]

出　版：協群科技出版社

發　行：協群科技出版社

香港中環卑利街684號二樓

印刷者：廣源印務局

青山道875號工廠大厦

定價：H.K.$ 25.00

目　錄

第一章 簡 介

這本書是爲補足蘋果公司的磁碟作業系統手册而寫。無論行家或生手，如想對磁碟結構做進一步的認識的話，都可從本書獲得更多的訊息。本書並不是要來取代原有的磁碟作業系統手册，但爲連續起見，原有手册已有部份亦將於本書重覆。又因不是每一個人都需用到本書所有資料，書內的每一章都可獨立閱談。

本書資料乃是作者及其它經體豐富的程式員對磁碟作業系統廣泛的分析及銓釋所得之結果，另外也有些是從應用手册，其它文章或與人討論而來。雖然不敢擔保這些資料完全正確無誤，但是它們已經過了徹底的研究及測試倒是可以斷言的。

有許多理由寫這本書：

1. 展示直接用組合語言存取磁碟作業系統之方法。
2. 幫助你修護完全無望的磁碟。
3. 修正及補充原有手册之不足。
4. 修改磁碟作業系統以適合你的需要。
5. 提供完整的磁碟格式。

當蘋果公司於1978 年提出她的第三版磁碟作業系統以配合它的DISK Ⅱ磁碟機時，幾乎沒有什麼註解附在上面。稍後磁碟作業系統（以下簡稱DOS ）3.2 發表時，才有一份 178 頁的指令及參



有許多理由寫這本書！

考手册問世，其中包含有DOS 的使用法（使用BASIC 語言）及包括一些DOS 內部操作情形。當DOS3、3來臨後DOS 3.2使用手册雖稍有更新但其資料仍然大體上甚無改變。除了上述這些手册及一些蘋果使用者雜誌與報導外，幾乎對DOS 之注解工作沒有多大改進。本書就是針對這種缺失而寫。

本書討論重點主要是在第 3.3 版的DOS 上，這是因爲 3.3版是最新的，且與 3.2版相差得非常之少，當遇到兩種版本有主要差異時，兩種版本都會被討論。

除了討論與DOS 有關的資料外，本書也討論了其它一些可應用在蘋果二世（ Apple Ⅱ ）及蘋果三世（ Apple Ⅲ ）的作業系統。在大部份情況下磁碟格式在各種作業系統上都是一樣的，軌道及段落的格式就是一例。

第二章 磁碟作業系統(DOS)之演進

蘋果DOS 自從問世以來已經歷經數次的修訂而有三個主要的版本。所有的版本在表面看起來非常的相像。在DOS3.3中找得到的命令DOS3.2及DOS3.1裏面通通也都有。因爲修訂的原因只是要修改DOS 內之錯誤及做少許的擴大而非著眼於增加其功能。只有在DOS3.3好容易才發現在功能上有一點主要的改進：一個軌道（ track ）所能容許的段落（ sector ）數目從13增加到16。

現來看看DOS 的歷史：

＊DOS 3　1978年6月29日問世

＊DOS 3.1 1978年7月20日問世

很明顯的，第一次推出的DOS 是爲了培襯DISK Ⅱ磁碟機而倉皇推出的，因此裏面錯誤一大把。爲了要迎合蘋果二世加強機種（ Apple Ⅱ. plus）的出世及自動開始ROM(AUTOMATIC ROM)元件的使用，作一次修改是勢在必行。

＊DOS 3.2　1979年2月16日出世。

雖然DOS 3.2號稱修改先前DOS 的程度比任何一次DOS 修訂的都多，但是90%的DOS3.1 還是被保留下來，構成了DOS 系統的基本架構，DOS3.1，DOS3.2及以後所提出之DOS 系統的主要不同點列於下面：

—NOMON C，I，O（命令，輸入及輸出爲非監督狀態）爲在DOS 3.2 系統下之自然値（default ），而MON C，I，O

(命令、輸入及輸出爲監督狀態)則爲在DOS3.1系統下之自然值。

—在DOS3.1下似若MON O狀態時，所輸入之催促信號(promp)(>,〕,*)將回應(echo)出來，而在MON I狀態時則無此回憶。

—當DOS命令從鍵盤輸入時，DOS3.1系統執行這命令完後會傳送一個空白及一個回車(carriage return)信號到BASIC程式上。而在DOS3.2系統執行命令完後則只傳送一回車信號而已。

—在DOS3.2系統中某些命令並不能從鍵盤輸入而只可寫在BASIC程式中(例如READ,WRITE,POSITION,OPEN,APPEND)。

—在DOS3.2系統中，當使用者想以RAM之格式處理他的程式時，系統可自動的把系統內本是ROM格式的程式轉換成RAM之格式以求配合，反之亦然。

—DOS3.1不能處理小寫字母而DOS 3.2卻能夠。

—DOS中某些命令可被允許來產生新檔，另外一些則否。在DOS3.1系統中，任何想要參考的檔不存在時則系統自動產生一個新檔，如此一來迫使DOS3.1把不再需要的新檔刪除掉，例如LOAD XYZ指令，如果XYZ檔不存在，則系統將自動產生它，刪除它，然後印出"檔沒找到"的錯誤信號。在DOS 3.2中則不然，OPEN命令被允許可以來產生一個新檔，而LOAD這命令則不行。

—DOS3.1系列中，從監督程式出來需把監督程式狀態暫存器(Status register)〔位於$48〕置爲零，然後才可再重新進入此監督程式。在DOS3.2中，這些通通不再需要。

—DOS 在執行讀／寫軌道／段落（R/W-Track /Sector）時，不接受外界中斷信號（ interrupt ），可是在DOS3.1系統中在作寫入磁碟的動作時可接受周邊裝置之干擾信號，但也會因此把磁碟資料弄壞。

– DOS3.2中B鍵語（即拜偏移量）若不指定將被置爲零。

—在DOS 3.2系統中DOS 程式會自動重新組合，使得各種副程式之位置會有些微改變。這對使用者來講也許是個災難，因爲他們可能把固定位址寫在他們的程式當中。

—DOS3.2中增了幾種檔案型（除了正規的T，I，A及B等形式外），但是還沒有使用這些新檔案型之命令問世。這些新型是S型、R型及新A型，新B型。只有R型已被用在DOS 工具鍵（ TOOL KEY）作爲可重定址目標模組合器檔。除此之外，在目前尚未用到這些新形式。

—在DOS3.2中有多附加自動起始（ AUTOSTART)ROM。

—在DOS3.2中當磁碟客滿時仍想OPEN一個檔，系統會自動把它關掉。

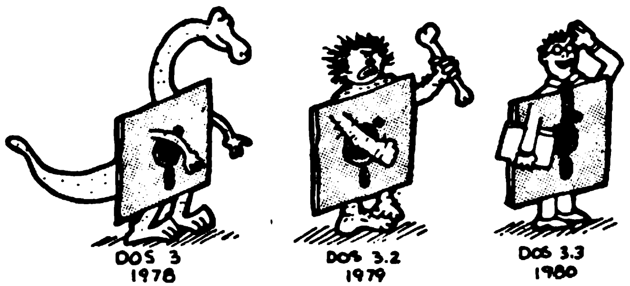
—正如其它修訂一般，DOS3.2也被加入了一些新程式。其中UPDATE 3.2程式被用來取代了原有的MASTER CREATE程式，其功能則是用來建一個主磁碟區。UPDATE 3.2 可把一奴磁碟轉換成主磁碟以及使HELLO 檔重新命名。

DOS3.2.1 1979年7月31日出世

DOS 3.2.1 主要是爲DOS3.2之維護目的而產生。對RWTS及COPY程式做了少許的修補以改進其時序問題，但也因此加入了一些延遲時間，使反應稍慢。

DOS3.3 1980年8月25日

DOS3.3在1980 年年中推出，被當作是DOS3.2.1在軟體及



蘋果磁碟作業系統的演進

硬體方面改進的版本。DOS3.3在控制器板上新加裝了靴帶式與狀態ROM 晶片，提供了讀、寫與格式化16段落式磁碟的能力。（這些ROM 與語言系統所用的ROM 相同），這種改進比以前13段式磁碟都省了25%的磁碟空間。DOS 系統在DOS3.3 使用手册中亦有所更新，BASIC 程式語言磁碟及主磁碟也被包括在內。雖然在此新系統內讀寫軌道段落這部份完全被重寫，但是DOS 系統其它部份並沒有被重新組合而且內容只有少許的修正：

—系統始初靴帶式輸入程式在DOS3.3中的位置被移爲$ 800，而在DOS3.2中則爲$300，除此之外，因爲程式在磁碟中被儲放於軌道0 及段落0，它也像其他段落似相同的分塊（nibbilized）技巧儲存。

—在APPEND 程式中，若檔的長度爲256 拜的倍數時，常常此檔會被不適宜的儲放。此缺失DOS3.3中已訂正過來了。

—在DOS3.3中執行完SAVE 及BSAVE命令後系統自動會執行一個查證（verify）命令以保正確。

—在內容表（Volume Table of Contents簡寫成VTOC）上記

錄空白段落位置共有四個拜（byte）都被使用了。這將使得DOS處理能力能達到每軌道32個段落。但由於RWTS硬體上的限制，只能夠達到每軌道16個段落。

—如要使用語言板（LANGUAGE CARD）時，DOS就在執行靴帶程式時將$E000位址資料設定爲零，這樣可使在主磁碟上之（HELLO）檔被重新輸入BASIC程式。

—DOS系統程式在DOS 3.3中讀進記憶體之方式爲從上向下而不是從下向上，但在磁碟中存放方式秩序（佔有軌道0,1與2的先後還是與一般相同。

—在DOS 3.3裏其它程式也可加入主磁碟檔中。如：

FID —爲一般化之檔案處理公用程式。它可使個別檔或許多檔被抄入磁碟中。

MUFFIN —爲一轉換程式，它可允許DOS 3.2的13段落檔被抄至DOS 3.3的16段落磁碟檔。

BOOT 13 —爲一可靴帶式起始13段落磁碟之程式。

及一新的抄寫程式，此一程式只需單一推動器抄寫。

—在DOS 3.2中，由於磁碟機速度不同，使得它們不能和DOS內抄寫程式合併使用。而在DOS 3.3中由於抄寫程式已重新修訂過，此項限制已不復存在。

第三章 磁碟格式

蘋果公司的磁碟作業系統只提供了非常基本的資料來說明如何安排磁碟格式。因此這一章裏我們詳細的來探討在一磁碟上如何來組織資料，第一節將簡略的敍述一下硬體，對此已熟悉之讀者可跳過此節不讀。

軌道及段落

爲了系統上的管理，DOS 把磁碟分成軌道及段落兩部份，在開始把資料放入磁碟就是如此分的。軌道的定義就是在磁碟上一條與磁碟中心同心的圓環狀軌跡。從軌道上與磁碟中心的距離我們可以辨別出不同的軌道，就像電唱機的唱針頭一般，磁碟的讀寫頭可被定位在任何給定的軌道上，這軌道有點像唱片上的溝槽，只是它們不是螺狀同心而已，在讀寫資料時磁碟就像唱片一樣以定速度旋轉，而讀寫頭就在磁碟上方讀出或寫入。蘋果磁碟共分有35個軌道，軌道 0 在最外邊，軌道 34在最裡面，圖31展示軌道的觀念，但實際上各個軌道在磁碟上是無法用眼分辨出來的。

正確的說，磁碟臂在指向軌道時分成70相（ phase ，如三相馬

達分成三個相一般。），所以要從一個軌道移到相鄰的另一軌道步進馬達需移動 2 相（ 75÷35＝2 ），這意味著若磁碟改成70軌而非 35 軌磁碟臂將仍能勝任，但是很不幸的，這樣做的話將使得讀寫頭鑑別率不足，常在兩磁軌間造成資料串音或損耗，標準的 DOS 系統在偶相時才有軌道存在，但也有些保護系統是在奇相或奇偶相混合使用，每個軌道必須相隔兩個相以上以保安全，有關保護系統更多資料請看附錄 B。

一段落就是一個軌道再分割出來的單位，當我們想更新資料時通常以段落為單位，DOS 每次都從一個段落讀出或寫入資料，這樣就可避免需要一大串記憶體當緩衝器（如果以軌道當基本單位的話則需要）。蘋果公司的磁碟現有兩種格式：每軌道13段落與每軌道16段落兩種。找出每個軌道第一個段落的方法並不像一般磁碟用索引洞（ index hole ）。這樣做的原因意味著軟體程式必需能夠自行定出軌道及段落所在而不需硬體來幫忙，這樣架構稱為軟體段落法，它雖然會浪費一些小儲藏空間但却富有彈性，這可由它能從每軌道13段落擴充到每軌道16段落得到證明，下表說明13段落及16段落之記憶容量。

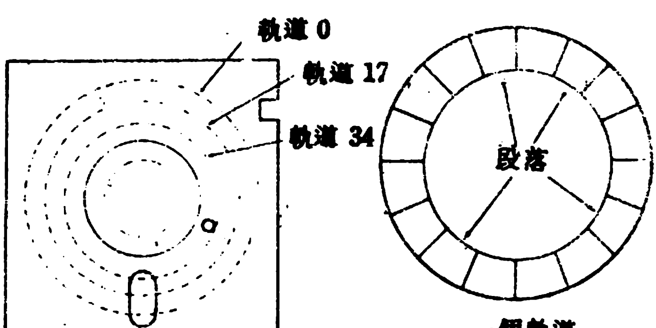


圖 3.1

磁碟結構

軌道數

所有DOS 版本……3.5

每軌道段落數

DOS 3.2.1 及以前版本……13

DOS 33 ……16

每個磁碟段落數

DOS 3.2.1 及以前版本……455

DOS 3.3 ……560

每段落所容拜（ byte ）數

所有DOS 版本……256

每個磁碟所容拜數

DOS 3.2.1 及以前版本……116480

DOS 3.3……143360

可使用的段落數*（做爲資料存放）

DOS 3.2.1 及以前版本…… 403

DOS 3.3…… 496

（*即扣除儲存DOS VTOC及目錄表所需空間）

每個磁碟所含可使用的拜數

DOS 3.2.1 及以前版本……103168

DOS 3.3……126976

軌道格式

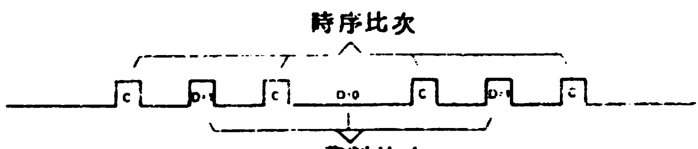
譯者註：本書此段所介紹的軌道格式爲舊式的5-5分塊（見本書3-15 頁），現今DOS3.3 所採的爲6-2分塊圖（見本書3-21 頁），其容量提高了50%。

到目前我們已解釋過軌道及段落之格式，現在再來探討資料如何被存取，我們從最基本的講起。

因爲本書主要講軟體，所以硬體規格說明往往就省略了，例如，資料是以連續類比方式被存放到磁碟，但我們卻要把它當作是分立的數位信號（即0與1）。只要知道有個硬體把類比信號變成數位信號即可，至於如何變法則不在本書討論範圍內。

在磁碟上的資料比次（bits）有一定的間隔，爲了討論方便起見，資料比次被畫成可拿時序比次來間隔，用這種表示法，資料都以圖3.2之形式讀出或寫入，圖中資料代表101。

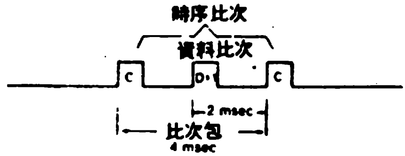
圖3.2　磁碟上的比次



從圖3.3可看出來，時序比數和資料比序是彼此交錯著，若兩時序比次間冒出一個脈衝則代表資料比次爲1，若沒有脈衝的話則資料比次爲零，現在定義一個名詞"比次包"（bit cell）：比次包爲一個時序脈衝昇起部份到下一個時序脈衝昇起部份所經過的時間，如圖3.3所示。

一個拜包含有8個比次包，值最大的比次包是比次包7，值最小

圖 3.3　比次包



的比次包是比次包 0 ，當我們說起資料比次時，就是在說它所對應之比次包。資料是以串接式被讀出或寫入的，一次一比次，因此在寫入動作時，比次包 7 最先被寫而比次包 0 則最後，同理在讀出時比次包 7 最先被讀出而比次包 0 則最後。下圖展示在一個拜內各比次間的關係。爲了圖示資料比次如何被存取，需要一些其他的表示法。上圖是發生在磁碟機內的，而在一個磁碟的硬體介面我們將以一個八比次的"資料閂"(data latch)來表示，從軟體的觀點來看，資料閂包含了資料流如何從磁碟上流入與流出的訊息。

圖 3.4　磁碟上的一個拜資料

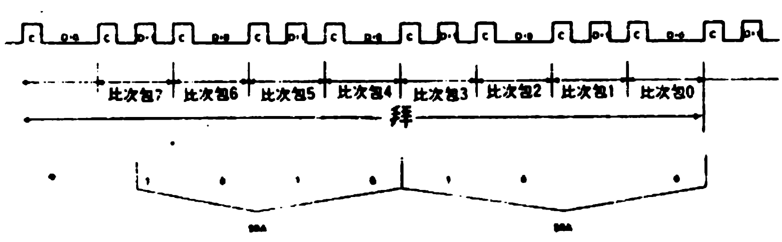


圖 3.5 顯示出三個比次 101 ，如何延磁碟內讀出至資料閂中，當然還有 5 個比次需讀入以填滿這資料閂，其次可看到資料比次料序比次所間隔，讀出的工作由硬體來執行。

寫入資料大致也一樣（看圖 3.6 ），在寫入一個資料比次時要

先插入一個時序比次脈衝。必須注意的是在"寫入"資料比次時，資料比次 0 實際是被讀入資料門內（置於最後，見圖 3-6）。

圖 3.5　從磁碟上讀取資料

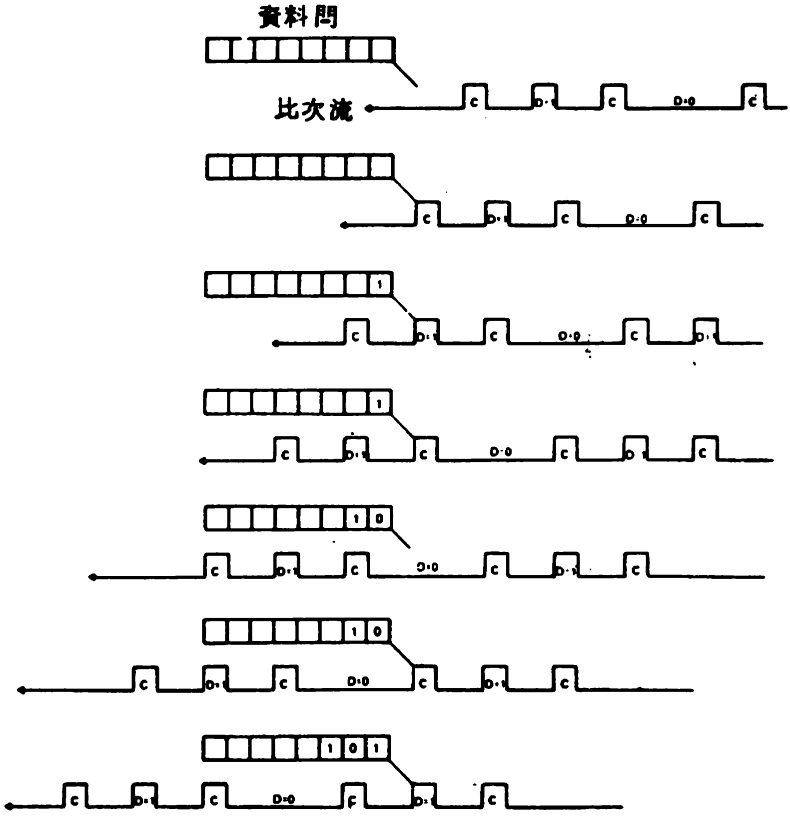
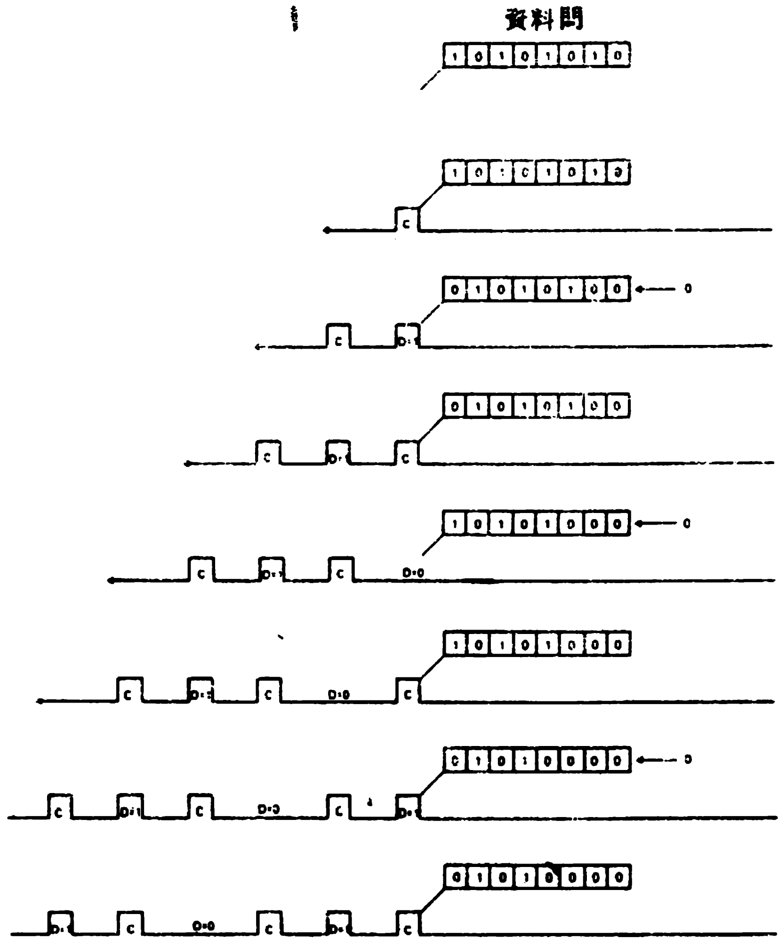


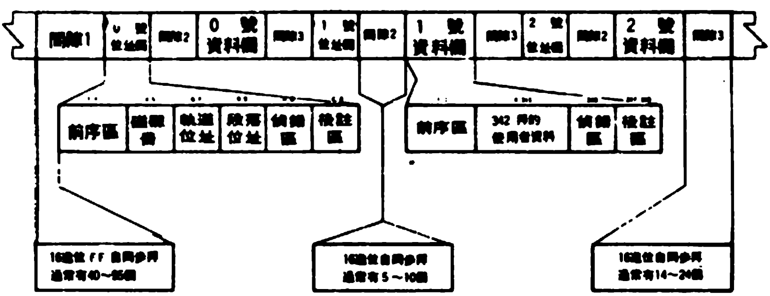
圖 3.6 把資料寫入磁碟



欄（ field ）是由連續的拜所組成，所含拜的數目可改變，視欄之性質而定。欄有兩種：位址欄與資料欄，這兩種欄資料結構頗類似，都包含有前序、資料區、偵錯區及後註等區。每個欄與相鄰欄都有幾個拜的間隔，叫做"間隙"（ gap ）。有間隔的好處有二，第一，它可使一個欄的更新不影響到鄰近的欄（在蘋果系統中只有資料區才需更新），第二，在資料欄還沒開始讀寫時，它給計算機時間來分析地址欄內之內容。

所有的間隙在內容上大體相同，它們都包含自同步16進位信號"FF"，而只有長度（所含的自同步拜數量）可能不同。圖 3.7 是一個典型的軌道分解成各個主要構成份子的圖。

圖 3.7 軌道格式



自同步（ self - sync ）或自動同步（ Auto - Sync ）拜是一種特殊的拜，在上圖中三個間隙中都含有它。爲什麼要這樣稱呼它就是因爲它能夠自動地使硬體與磁碟上的資料拜產生同步。又爲什麼需同步呢？在前面我們曾前提到磁碟上資料比次是以連續的方式被排放，我們無法知道一個拜的 8 個比次從何處開始及何處結束，一個比次包和它的鄰居相同。當一磁碟推動器被指示來讀資料時，在一個

指定的軌道上資料將從任何一個位置讀起，也就是說在一個軌道上大約 50,000比次的資料中那一個比次都有可能被最先讀出。辨清時序比次及資料比次後，硬體先找出拜的第一個比次包然後再續讀7個比次包好放入資料栓當中。在上面的敍述我們假定可找到相當於每個資料拜的第一個比次包，其實，開始拿出的比次包可能是拜內八個比次包內的一個。圖 3.8 所示的是在一個軌道內之資料，其中時序比次已被移走。

圖 3.8 磁碟上比次流之一例

0 1 1 0 1 0 1 1 1 0 1 0 1 1 0 0 1 1 1 1 0 1 1 0 1 1 1 0 1 0 1

從上面我們看不出什麼樣的拜來，因爲我們不知道那裏是拜的第一比次。而這正是自同步拜所要克服的困難。

一個自同步拜被定義爲一個16進位數字 FF再在尾巴加上兩個零，所以事實上它是個10比次的拜。圖 3.9 展示自同步拜和磁碟上其它正常的拜結構的不同。

一般拜的 16 進位FF數	自同步拜 16 進位 F F 數
1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1 0 0

圖 3.9

自同步信號是以一40週期（微秒）爲單位重覆的產生自同步拜。因此每個比次需要 4 週期的時間。而兩個零也被讀入資料栓中，當 FF 被寫入磁碟時這兩個零也一併被寫入，成爲 10 比次拜。（DOS 3.2.1 及以前版本是使用 9 比次拜，因爲它們的硬體不能偵測

出兩個零來。）由圖 3.10 可知，只要5個自同步拜即足夠保證取得有效資料。理由如下：由於硬體需要一個拜的第一個比次爲1，當開始讀時，圖 3.10 上方即5個自同步拜，而下面幾列即表示硬體從第一個拜的任何一個比次開始讀，在各種情況下，當硬體至多歷經 4個自同步拜後，它就有能力與接著而來的拜產生同步，而把資料拜一個接著一個正確的讀出，除非磁碟存放的資料有錯。

現在我們來詳細的討論磁碟各個部份。首先來討論軌道內的三種間隙。這間隙長度與其它磁碟格式不甚相同，它們會隨著軌道不同，磁碟機不同而發生長度變化。在起始狀態下，DOS 將造出些較大的間隙然後使長度漸漸的減少，直到再少下去資料就會重覆的最佳狀態，這最佳狀態起碼要包括有5個同步拜才可。在軌道內部間隙長度就都趨於一致。

5個自同步拜

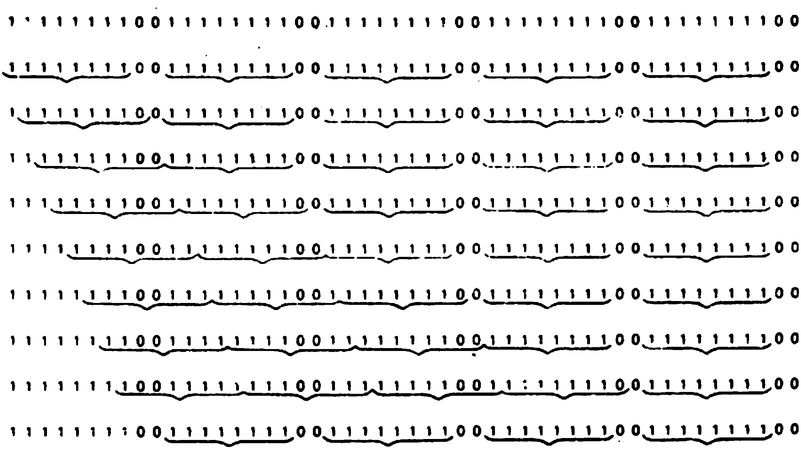


圖3.10

間隙1（見圖3.7）是在起始時第一個寫到軌道上的資料。它的任務是雙重的。在開始時這個間隙含有128個自同步拜，它可保證這整個軌道都可放滿資料（下面分解）。由於磁碟機的速度不一，軌道內長度使用率及每個應所佔長度都是未知數。所以在起始時先安排放一個間隙1，雖然由於磁碟機速度不一我們預測不到什麼時候軌道被寫滿，但是當間隙1開始被繞了一圈後的資料蓋上時，我們即可知軌道資料客滿。間隙1不能全被資料蓋掉，它起碼要有間隙3（下述）的長度（起碼要40個自同步拜），現在它的功能就轉變成間隙3的形態，其下個位址欄就是第一個位址欄（見圖3.7）

間隙2是在位址欄後插入而在資料欄之前，它的長度約5到10個拜（對正常磁碟機而言）。它的主要功能是給計算機解碼時間（當計算機已收到位址欄而還沒開始處理資料欄前）以便在收到資料欄時產生動作。如果間隙過小，可能發生資料欄已溜過而DOS仍在那裏考慮是不是這個段落要被讀取，6個自同步拜所產生的240週期似乎太足夠來解碼一個位址欄。當一個資料被寫入之時，又有

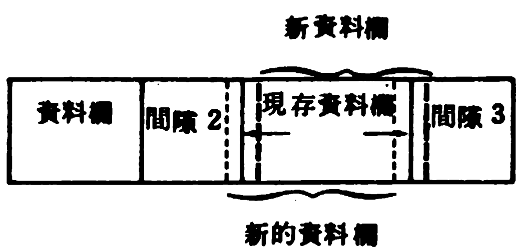


圖 3.11

問題發生，我們無法保證寫入的資料就是原來要被更改的地方，因爲磁碟機速度會變，所以寫入的動作也許會從一個拜的中間開始（見圖 3.11 ）只要不離譜太遠這還不成問題，我們在寫入資料欄之前可在間隙 2 先寫入 5 個自同步拜。這有兩個目的，首先，由於解碼位址欄所花時間不多，而去寫這 5 個拜可使資料欄更接近於它的起始位置。第二，更重要的，在讀的時候這 5 個拜就可被當作讀同步化信號。又在寫入一個資料欄時極可能至少要破壞掉一個自同步拜，因爲寫自同步拜時不太可能發生在原自同步拜的起始位置而使得一個拜必須遭到破壞，圖 3.12 顯示出這些。

靠自同步拜的幫忙寫入資料

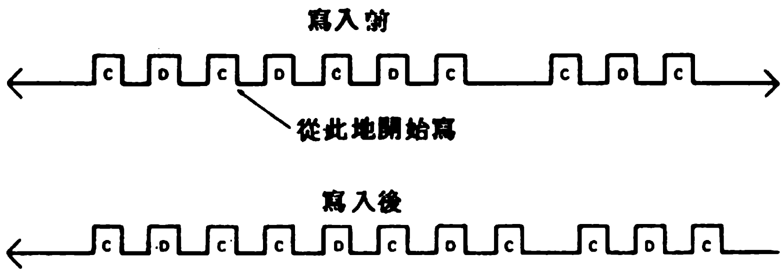


圖 3.12

間隙 3 是在每一個資料欄之後而在每個位址欄之前。它比間隙 2 要長，通常長度在 14 到 24個拜之間。它存在的目的和間隙 2 頗類似。間隙 3 使得計算機有時間處理收入的資料而然後有能力處理下一段落。間隙 3 的長度並不像段落 2 那般重要。如果下一個位址欄不是馬上來到，DOS 就在那裏等一直到出現爲止，當然它頂多不會超過磁碟轉一圈所需的時間。因爲位址欄從不被重寫，所以間

隙 3 也不必提供同步的設備。而本間只有前部會被前一資料侵入（見圖 3.11 所示）。

現在開始我們要來看看地址欄及資料欄內所含何物，位址欄裏面含的東西是位址或是一些有關其後資料欄之辨識資料。一 段落可由它所屬之磁碟册、軌道、段落編號而找到，正有如一棟房，可由國家、市、門牌號碼來找到一般。圖 3.7 已顯示位址欄內有很多元素，圖 3.13更清楚的展示這些：

位　址　欄

前序區	磁碟册	軌道	段落	偵錯區	後註區
D5 AA 96	XX YY	XX YY	XX YY	XX YY	DE AA EB

奇一偶　編碼

資料拜 — $D_7D_6D_5D_4D_3D_2D_1D_0$
XX — 1 D_7 1 D_5 1 D_3 1 D_1
YY — 1 D_6 1 D_4 1 D_2 1 D_0

圖 3.13

前序包含有 3 個拜，含有一獨一無二的序列。這使得DOS 能夠正確無誤的識別出它是位址欄來。此三個拜分別爲 $D5，$AA，及 $96，$D5及$AA 是保留拜（資料內不准有這種編碼）就是這樣使得前序獨一無二。$96指出跟於其後的資料構成了一個位址欄的欄的資料（而非資料欄資料）。跟著而來的就是磁碟册、軌道、段落編號及偵錯區，這些資料是絕對須要的來使得 DOS 知道它現在那一塊磁碟的那一個位置上，偵錯區是把前三項資料做互斥或（exclusive OR）而得，目的是證實資料無誤。最後是後註，它包含

有三個拜：$DE、$AA，及 $EB 後註拜有時被稱作比次失誤記號器因爲它可確定磁碟機與磁碟上的資料是否仍同步。這些拜也許可以不要，但它們也提供了再檢查一遍資料的方法。

另外一種欄就是資料欄。和位址欄構造相同，它包括前序、資料、偵錯區及後註（參考圖3.14 ）。前序只有第三個拜和位址欄不同，這些拜是 $D5，$AA，及 $AD，也構成了一個獨一無二之序列，使得DOS 能辨識出一個段落之開始。資料區包含有 342拜的編碼資料，如何編碼將在下節解釋。接下來是偵錯區（只有一個拜），用來查明剛讀入的資料是否正確，後註區的功能和位址欄內的完全相同。

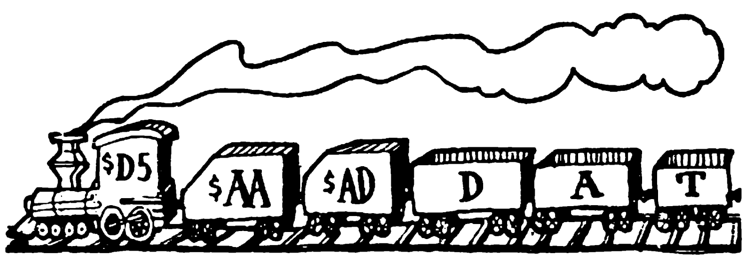
資 料 欄

前序區	使用者資料	偵錯區	後註區
D5 AA AD	342 BYTES DATA	XX	DE AA EB
	6 之 2 編碼		

圖 3.14

資料欄編號

由於蘋果公司的硬體限制，不可能一次從磁碟上讀出一個段落其中所有可存放的 256 拜都是合乎硬體要求的需要。這個問題並不難解決。事實上在寫入時資料需要經過編碼。有三種不同的方法。



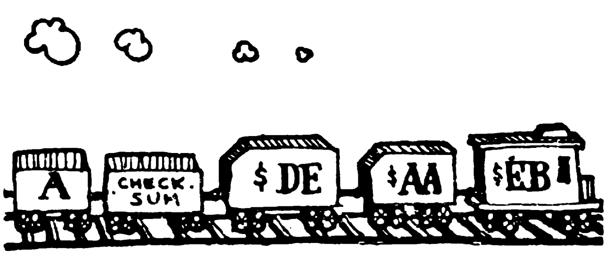
資 料 欄 編 碼

第一種，現在被用在位址欄上，是把一個資料拜分成兩個磁碟拜，其中一磁碟拜包含奇數比次，另外一個則包含著偶數比次。因此這方法需 512 磁碟拜來存放一個段落所含的 256 個資料拜，如此一來，一個軌道將裝不到10個段落，這樣相當於每個磁碟可存88K 比次資料，而大約有72K 比次可供使用者來用，這樣對一個 5 ¼ 英吋的單密度磁碟機來說是很標準。

幸運的，第二種寫資料進磁碟的技術被發明了。它可使每個軌道寫到13個段落。此法乃是把一個拜分割成 5 次次及 3 比次而非第一法的 4 比次及 4 比次方式，寫到磁碟的拜中有 5 個比次是有效的資料而不是 4 個。如此一來只要 410 個磁碟拜就可儲存一段落所含的 256資料拜。這種密度剛好每個軌道內可被放入十三個段落，這種形式被用在 DOS3 到 DOS3.2.1 之系統當中而比現今其它類似的磁碟密度提高了33% 之多。

現在 DOS 3.3 硬體每個軌道已放入16個段落，因此比第二法更提高了 3.2 % 的密度。這是由改進硬體而達到的（在磁碟控制板上使用了 P6 PROM），此硬體容許了一個拜分開成 6 比次與 2 比次，這些都是由於存於 P6 PROM 的自動機邏輯來達成的。它容許了兩個連續的零可同時存在於一個資料拜中。（亦即以前硬體做不到這點）。

現在我們更詳細的來看看這三種不同的編碼技術。DOS 3.2.1



及其以前的DOS 的硬體對資料的存與取有如下幾項限制：它要求一個磁碟拜值最大的比次爲 1 且不能有連續兩個零的資料存在於一個磁碟拜上。第一法的奇數比次與偶數比次分開 4 比 4 的技巧符合了上述的要求。一個資料拜被分成兩個磁碟拜而未使用的比次全部爲 1 ，如圖 3.15 所示。

資料拜 $D_7D_6D_5D_4D_3D_2D_1D_0$ → $1D_7\,1D_5\,1D_3\,1D_1$
　　　　　　　　　　　　→ $1D_6\,1D_4\,1D_2\,1D_0$

圖 3.15

不管是任何資料拜，這方法都保證最大的比次爲 1 及沒有連續兩個零比次發生。此法被用在位址欄中的册及軌道段落編號資料。要解碼這些資料也不難，只要把奇數比次拜的各個比次向左移，然後再與偶數比次拜的各個相關比次作AND 的運算，如圖 3.16 所示，原資料拜就重現了。

$$
\begin{array}{ll}
 & D_7\,1\,D_5\,1\,D_3\,1\,D_1\,1 \quad \text{左移一位} \\
\text{AND} & \underline{1\,D_6\,1\,D_4\,1\,D_2\,1\,D_0} \\
 & D_7D_6D_5D_4D_3D_2D_1D_0
\end{array}
$$

圖 3.16

須注意到奇數比次拜被左移時值最小的比次要被塡入1。這整個運算是由放置在DOS 系統（ 48K ）$ B 944 的地方之RDADR副常式來完成的。

上法的缺點是太浪費軌道的空間，爲了克服此點，" 5 對 3 " 的編碼技巧就被發明出來了。它被如此命名是因爲它把一個拜分成 5 個比次與 3 個比次兩段。而一個資料拜先被分割成如下之格式：

000×××××，×是一個有效的資料比次。上面這種格式的拜範圍爲$00到$1F，總共有32種變化。而從$AA到$FF間的範圍內，總共有34個磁碟拜符合最大比次爲1及沒有兩個連續零的條件。）把$AA及$D5兩個拜保留下來後，剛好每個分割後的資料拜可對應到一個合乎磁碟硬體要求的磁碟拜中，因此，把一個資料拜轉換成磁碟拜是經過兩重手續的，圖3.17顯示出這些關係。

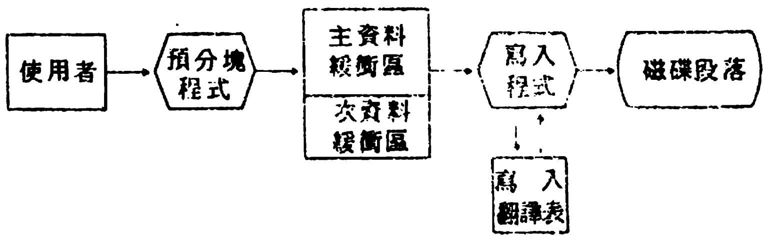


圖 3.17

首先，構成一個段落的256個資料拜必須首先被轉變成5比次拜，這是由一位於DOS $B800位置的預分塊（prenibble）副程式來執行的。這是一個相當複雜的程序，包含有許多比次重新排列的步驟。在最左邊是一個資料儲存區，裡面的資料拜被DOS傳給RWTS副程式，見圖3.18每一個拜由一個字母來代表（如A, B, C等），拜內每一個比次則加數字註脚（7到0）。右邊緩衝區是轉換後的結果。其中主緩衝區包含有5個不同區域，每個區域已有5比次有效資料拜（最大的前三個比次被零所填滿）。第二緩衝區包含3個不同的區域。圖如其名的顯示出"5比3"之態勢來。

總共需要有410拜的緩衝區來存放原有的256拜資料。這可計算如下：比次總數256拜×8比次／拜＝2048比次。而在緩衝區每個拜的有效比次爲5，所以2048比次÷5比次／拜＝409.6拜

5 及 3 分塊法

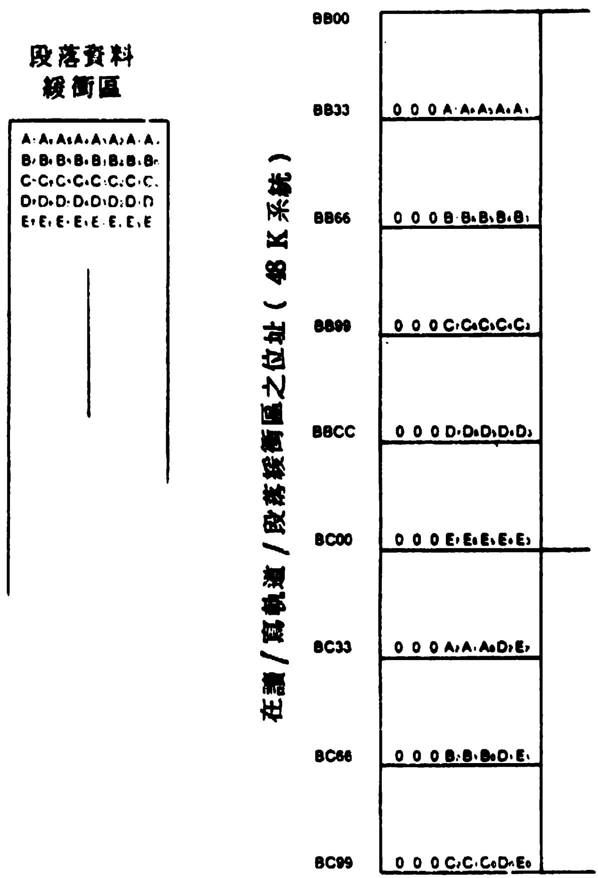


圖 3.18

，（兩個比次沒用到）。當這步驟完成後，這些緩衝區的資料更進一步被對應轉換成合乎磁碟要求的磁碟拜。這一步倒是很簡單，只要一個表對應過去就可以了，見圖 3.19 。

5及3分塊法的寫入翻譯表

00 = AB	10 = DD
01 = AD	11 = DE
02 = AE	12 = DF
03 = AF	13 = EA
04 = B5	14 = EB
05 = B6	15 = ED
06 = B7	16 = EE
07 = BA	17 = EF
08 = BB	18 = F5
09 = BD	19 F6
0A = BE	1A = F7
0B = BF	1B = FA
0C = D6	1C = FB
0D = D7	1D = FD
0E = DA	1E = FE
0F = DB	1F = FF

AA
D5 } 保留拜

圖3.19

資料欄內有個偵錯區，和位址欄內的偵錯區很類似。它是被用來證實資料有沒有錯誤。它也是由資料之互斥或而得，但由於讀出時間的限制，作這偵錯區的方法稍有不同。資料重覆成對的被互斥或後才經查表轉換成磁碟拜。圖3.20*顯示了這些。

*圖3.20及圖3.21編碼方式也許不同（5對3及6對2）但它們從緩衝區到磁碟所經歷的過程則完全相同）。

爲何要如此轉換的理由將會很明顯如果我們進一步考慮資料如何從磁碟中被讀出的話。讀出程式必須先讀一個拜，然後轉換它和儲存它，所有這些必須在32週期內完成（這時間也是寫入一個拜所需的時間）否則資料將會遺失，用上述那種偵錯區計算方式來解調資料，由圖3.20 可以看出來，這種方式充分的利用了時間而可在時間限制下達成任務。當資料正被從一個段落讀出來時，累積器（

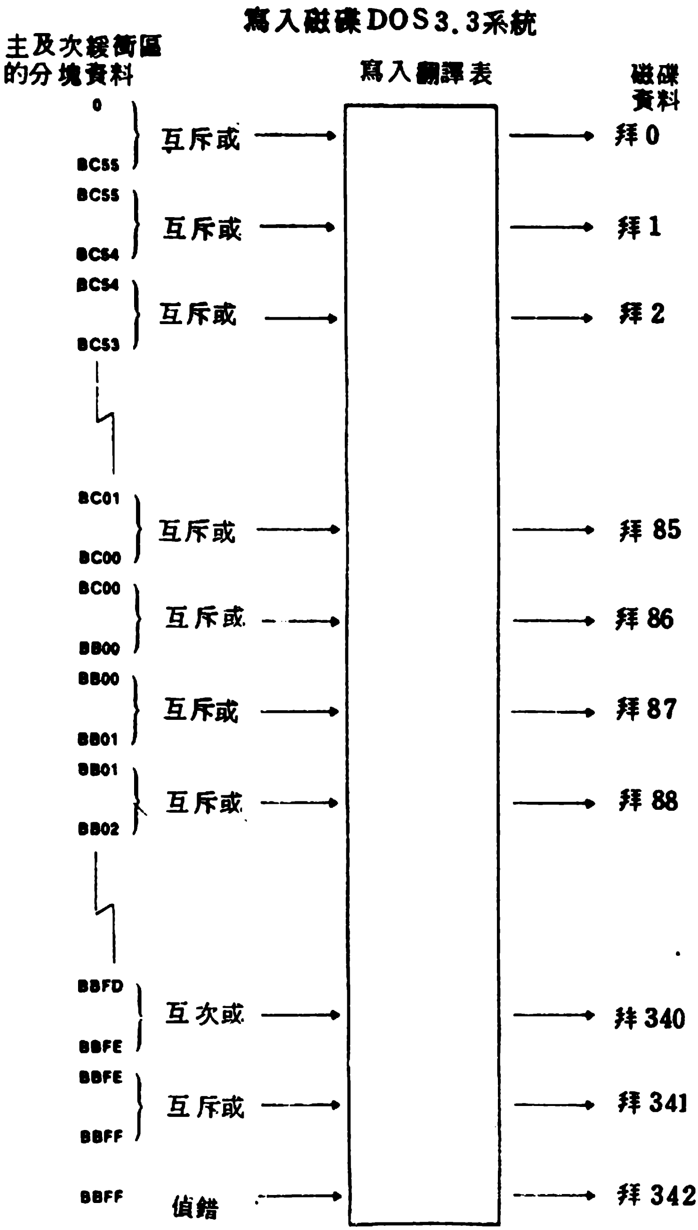

寫入磁碟DOS3.3系統
主及次緩衝區
的分塊資料
寫入翻譯表
磁碟
資料
0
BC55
互斥或
拜0
BC55
BC54
互斥或
拜1
BC54
BC53
互斥或
拜2
BC01
BC00
互斥或
拜85
BC00
BB00
互斥或
拜86
BB00
BB01
互斥或
拜87
BB01
BB02
互斥或
拜88
BBFD
BBFE
互次或
拜340
BBFE
BBFF
互斥或
拜341
BBFF
偵錯
拜342

圖 3.20

從DOS3.3磁碟讀出

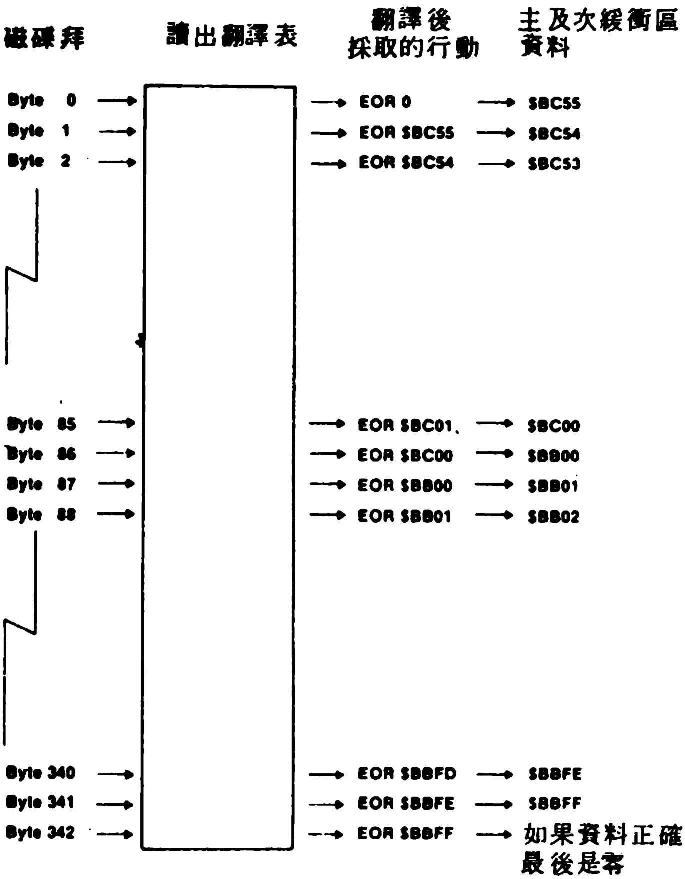


圖 3.21

accumulator ）正含有所有以前拜作互斥或後的結果。從轉換表出來的資料和累積器資料作互斥或後所得結果正是所需的資料。這過程圖示於圖 3.2.1 。

第三種編碼方法現已被 DOS3.3 使用，它和" 5 對 3 " 方式非

常類似。它的產生是由於硬體有所改進而對磁碟拜的要求不再那麼嚴格。拜內最大的比次仍需為1，但是一對連續的零資料（也僅可有一對在一個拜中）已可容忍。這樣一來就可以使一個拜內有效比次增加而產生了" 6對2 "的編碼方式。一個含有6位有效比次的拜格式如下：00××××××，其值從$00到$3F，總共有64個不同的值，以上述對磁碟拜放寬標準的限制來看，從$96（第一個合標準的）到$FF 總共有69個合乎限制的要求。除去兩個系統保留拜$AA及$D5 外，還有67個，而需要量僅為64個。好！再加它一個限制，此即一個拜內至少必須有一對1連在一起（但和比次7的1連在一起的不算）。如此一來，恰成64個對64個，預分塊（preniblized ）後的資料拜和磁碟拜恰成一對一的關係。起始轉換仍是由DOS 系統位於$B800的預分塊程式來執行，其結果在圖3.22。

在緩衝區需342拜來寫入一256拜之段落，計算如下：

256拜×8比次/拜=2048比次

2048比次÷6比次/拜=341.33拜。

從6位有效比次拜（在緩衝區內）轉換成有效的磁碟拜中間的一對一關係列於圖3.3。和" 5對3 "一樣，寫入磁碟內之資料拜也要經過互斥或閘處理。

段落交錯法

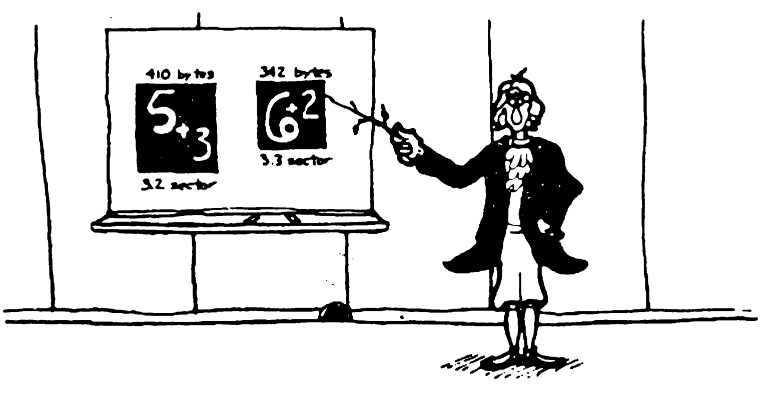
段落交錯法即是在一個軌道中把段落不按次序任意放的一種技巧。它可使取用速度提高。DOS 在讀與寫一段落完後要跳到另一

" 6 及 2 " 分塊法寫入翻譯表

00 = 96	10 = B4	20 = D6	30 = ED
01 = 97	11 = B5	21 = D7	31 = EE
02 = 9A	12 = B6	22 = D9	32 = EF
03 = 9B	13 = B7	23 = DA	33 = F2
04 = 9D	14 = B9	24 = DB	34 = F3
05 = 9E	15 = BA	25 = DC	35 = F4
06 = 9F	16 = BB	26 = DD	36 = F5
07 = A6	17 = BC	27 = DE	37 = F6
08 = A7	18 = BD	28 = DF	38 = F7
09 = AB	19 = BE	29 = E5	39 = F9
0A = AC	1A = BF	2A = E6	3A = FA
0B = AD	1B = CB	2B = E7	3B = FB
0C = AE	1C = CD	2C = E9	3C = FC
0D = AF	1D = CE	2D = EA	3D = FD
0E = B2	1E = CF	2E = EB	3E = FE
0F = B3	1F = D3	2F = EC	3F = FF

AA
D5 } 保留拜

圖 3.23



很簡單 - "6 及 2 " 分塊法比 " 5 及 3 " 分塊法要佔有較少磁碟空間

6及2分塊法

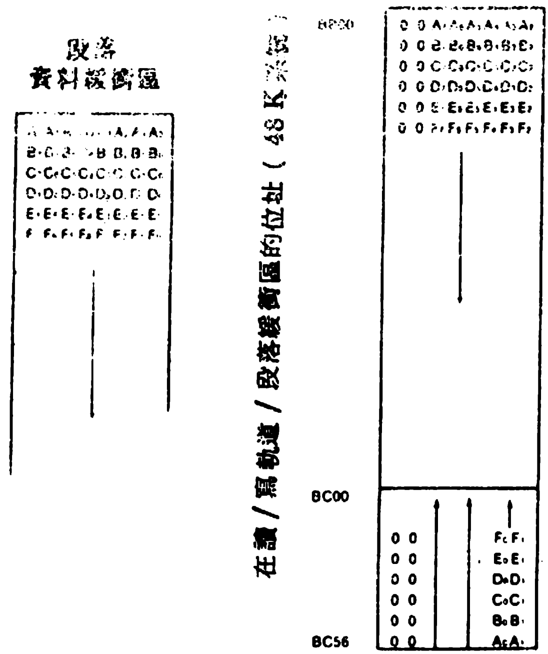


圖 3.22

段落時會有一段遲延時間。這遲延時間和應用程式如何利用磁碟有關而且相差很大。如果段落在磁碟內按著次序從小排列大，那麼除非應用程式反應很快，不然通常要轉一整個圈才能找到第二個要取用的段落。若是能夠把這些段落用不同次序或者是交錯著排列，那麼取用速度就將有所不同。

DOS3.2.1及以前的系統，磁碟13個段落是採交錯式的排列。因爲DOS 在磁碟上存放次序的段落編號是從小到大，而檔案系統存放次序的段落編號是由大到小，採單一的交錯式架構並不能同時

使軌帶式引進程式（ booting ）及次序的取用檔案都作得很好。

DOS 3.3 採用了另一種不同的方式來達到最高的速率。交錯式段落現在由軟體來做。16個段落以編號小到大（ 0,1,2 … 15)的次序放入磁碟中，其中一點交錯也沒有。然後有一個表把這些實際段落編號映射到另外一組假的或軟體的段落編號以供DOS 來使用。舉例來說，若是在磁碟上的段落編號爲 2（實際編號）查下圖得知其對應到軟體編號 $OC (11)，這時DOS 把實際編號爲 2的段落當作其編號爲11來使用。如果系統用RWTS來取用磁碟的話，這樣做一點問題也沒有，但如果系統沒有RWTS 的話就必須要小心了。

爲了消除在叫入起始程式及讀一些檔案時取用的時間的不同，我們做了另一修正。在叫入起始程式的過程中，DOS 系統程式是被倒著裝載進去磁碟，由段落編號大的地方開始裝，正如檔被裝載到磁碟上一般。這樣一來，只要一種交錯式架構即可達到取用磁碟資料時間最短的目的。

PASCAL FORTRAN 及 CP/M 等之 OS磁碟也用這種軟體交錯式架構，但是交錯的方式不盡相同，圖 3.24 顯示了這些不同。

段落交錯法之比較

實際磁碟的段落	DOS3.3 段落	PASCAL 段落	CP/M 段落
0	0	0	0
1	D	2	3
2	B	4	6
3	9	6	9
4	7	8	C
5	5	A	F
6	3	C	2
7	1	E	5
8	E	1	8
9	C	3	B
A	A	5	E
B	8	7	1
C	6	9	4
D	4	B	7
E	2	D	A
F	F	F	D

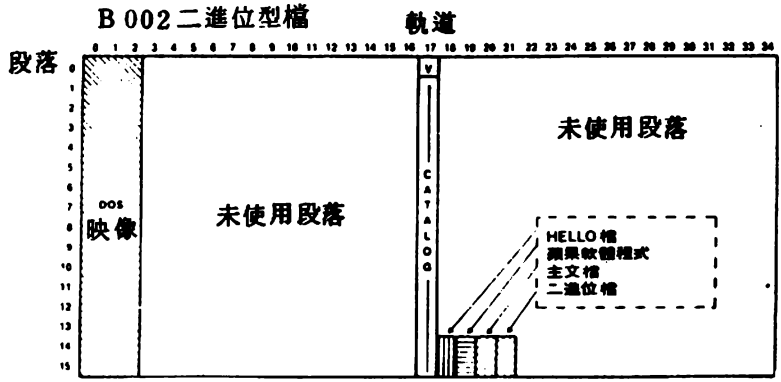
第四章 磁碟之組織

如第三章所述，每軌道16段落的磁碟共可容有 560 段落，每段落 256拜。這些段落分佈在35個同心的軌道中。本章目的的就是要說明DOS 系統如何安置這些軌道與段落。

一個檔（不管是蘋果軟體檔、整數檔、二進位數檔或主文檔）都包含有至少一個或數個的段落來儲存資料。因爲段落是磁碟上最小的單位。一個檔起碼要用掉一個段落，即使這個檔不滿 256拜。不滿的部份就浪費掉。因此，若一個檔含有 400拜的資料，它將填滿一個段落及把另一段落填 144拜，留下 112拜白白浪費。到此，我們就會想，有沒有辦法把磁碟上所有 143,360拜〔16段落/軌道×35軌道/磁碟× 256 拜/段落充分管理利用。事實上磁碟上能存的最大檔只 126,000 拜長。爲何如此就是因爲有些地方必須用來儲存控制程式。

控制程式段落包含有DOS 系統程式（它是在執行磁碟起始程式時就被裝載進來的），磁碟上檔的位置表及名稱表，還有磁碟上空白段落位置的記憶等等。DOS 如何利用磁碟圖示於圖 4.1 。

目 錄
磁碟冊 001
I 002 HELLO檔
A 002 蘋果軟體程式
T 002 主文型檔
B 002 二進位型檔



一個標準 16 段落型磁碟圖

圖 4.1

磁碟空間安排

圖4.1 顯示磁碟的前三個軌道永遠保留給DOS 從記憶體移來的程式映像。最中央的軌道（軌道17）放置VTOC及目錄。把目錄放置在中央的理由是很簡單的。因爲使用磁碟最大的延遲時間就是等待讀寫頭從一個軌道移至另一軌道移至另一軌道，所以若能儘量減少這些讀寫頭的移動將是非常好的。把目標放在最中央的軌道。這樣一來其他軌道要參考目錄軌道時最大移動就不會超過17個軌道。當檔被載入磁碟上時，首先就是從目錄軌道後的第一個軌道（第18軌道）開始塡起，一直塡到軌道34，然後再從軌道16繼塡， 15,14軌道一直下去到撞到存放DOS 系統程式映像時爲止。如果磁碟上僅存有很少的檔時，這些檔就會聚集在目錄軌道附近，這樣讀寫頭

所做的運動就可減少。每個檔都也可在自己所在之軌道上預留些空間以備以後使用。當一軌道塡滿時，就照上列方式再找一個空軌道來用。

册內容表VTOC

VTOC是整個磁碟之錨，任何DOS 所可取用的磁碟其VTOC都在同一位置：軌道17，段落 0 。因爲檔可在磁碟上任意地方開始或結束，DOS 系統就是藉著VTOC表來找出各檔之位置。VTOC有如下之格式（拜偏移量，以16進位表示）：

拜	內容
00	未使用
01	目錄軌道的第一個段落所屬之軌道號數
02	目錄軌道的第一個段落之段落編號
03	用來使磁碟起動的DOS 版本號數。動。
04～05	未使用
06	磁碟册號數（ 1～254 ）
07～26	未使用
27	軌道段落表段落所能編入之"軌道段落對"之最大號數（對 256 拜段落來說其值爲 122 ）
28～2F	未使用
30	最後段落被安置之軌道位置。
31	軌道配置之方向（ ＋1 或－1 ）。
32～33	未使用
34	每磁碟上軌道數目（正常情形下爲35）

35	每軌道上段落數目（13或16）
36～37	每段落之拜的數目（先底拜後高拜格式）
38～3B	在軌道零之未使用段落之比次對映(BIT MAP)
3C～3F	在軌道1之未使用段落之比次對映
30～43	在軌道2之未使用段落之比次對映
…………………………	
BC～BF	在軌道33之未使用段落之比次對映
C0～C3	在軌道34之未使用段落之比次對映
C4～FF	若未來每磁碟有35個以上的軌道這些拜就保留給它們用。

在一給定之軌道上未使用段落之比次對映

每一個軌道我們用32比次來表示軌道內段落之使用情形。16進位的段落編號如下所示地與比次位置對映：

拜	段落編號
+0	FEDC BA98
+1	7654 3210
+2	•••• ••••（未使用）
+3	••••••••（未使用）

因此，若只有段落E及段落8未使用則比次對映爲：41000000（16進位），若全部段落都未使用則爲：FFFF0000（16進位）（此即未使用之段落在比次對應上爲1，否則爲0）。

圖4.2就是以圖4.1之磁碟構造對映出來之VTOC段落。

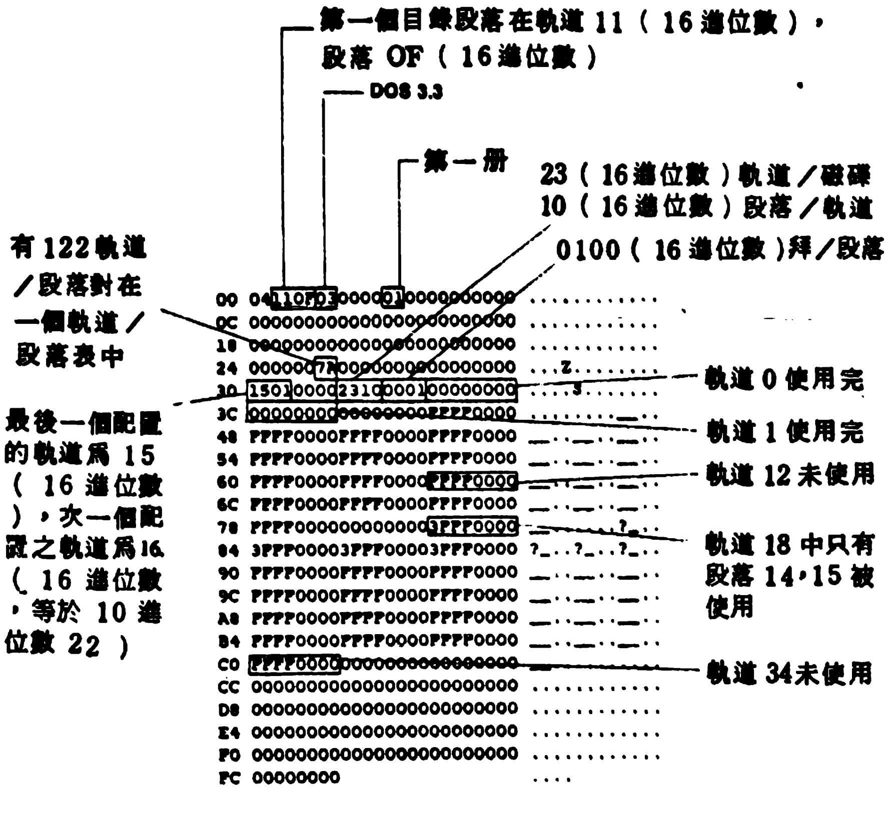


圖 4.2 一册內容表之例

目錄段落

爲了使DOS 能找到一個指定的檔，首先必須去讀VTOC以找出第一個目錄段落位於何處。一般，目錄段落就是把軌道17，VTOC部份除去後所剩下的其餘段落。當然，只要軌道及段落指標

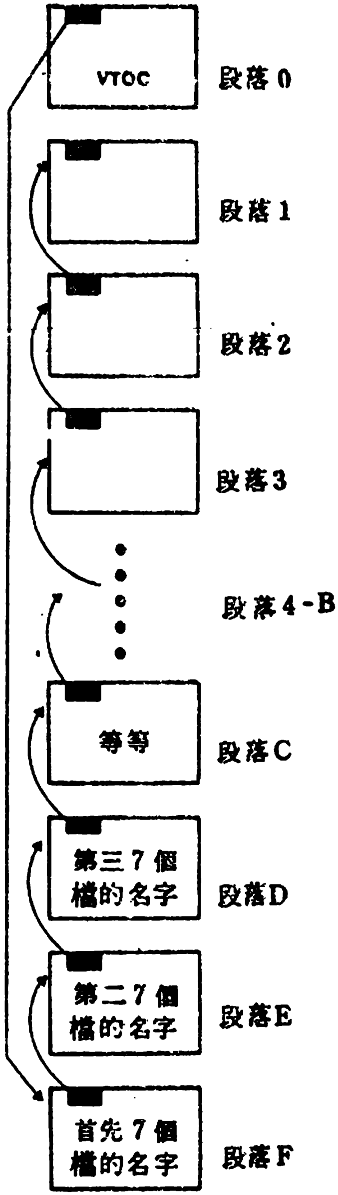


圖 4、3 — 軌道 17，目錄軌道

存在於VTOC中而VTOC也是位於軌道17及段落 0，DOS 就不需去管目錄位於何處了。圖 4.3 顯示出目錄軌道之結構。由圖 4.2 知VTOC中拜 01 及拜 02指出該目錄第一個段落位於軌道17、段落 F，目錄軌道的最後一個段落是一個目錄段落，它包含了磁碟最前面的 7 個檔的索引。每一個目錄段落和VTOC 一樣在拜 01 及拜02有一個軌道及段落指標指向下一個目錄段落。而目錄段落 1 的軌道及段落指標則為 0，表示終點到了。

在每一個目錄段落內可以容下七個檔的索引描述。由此，在 DOS 3.3 磁碟內可以有 7 檔（每目錄段落）× 15個目錄段落＝105 檔。一目錄段落之配置如圖 4.4 所示，其中

拜	內容
00	未使用
01	下一個目錄段落之軌道號數（經常是17）
02	下一個目錄段

落之段落號數。

03 ~ 0A	未使用
0B~2D	第一檔之索引進入點
2E~50	第二檔之索引進入點
51 ~ 73	第三檔之索引進入點
74 ~ 96	第四檔之索引進入點
97 ~ B9	第五檔之索引進入點
BA~DC	第六檔之索引進入點
DD~FF	第七檔之索引進入點

檔案索引進入點格式

相對拜位置	內容
00	第一個軌道及段落表所在的軌道。如果檔被刪除，此拜就變成一16進位數FF 。原來檔的軌道數目被抄到檔名稱欄（拜20）中最後一個拜上。如此拜包含一16進位數00則表示此進入點從未被用過且等著被使用。（此表示眞正的軌道0從未被使用來儲存資料檔，即使實際上DOS 之系統程式映像已從軌道0中被掃除。）
01	第一個軌道及段落表所在的段落。
02	檔型及旗號 16進位數80十檔型號數＝檔被鎖制。 16進位數00十檔型號數＝檔未被鎖制。 檔型有：

00　主文檔

01　整數BASIC 語言程式檔

02　蘋果軟體BASIC 語言程式檔

04　二進位數檔

08　S型檔

10　可重定位目標程式模組檔

20　A型檔

40　B型檔

由上可知，檔型號數 84 代表鎖住的二進位型檔，90 則代表鎖住之可重定位目標程式模組檔。

03 ~ 20　檔名稱（30個字母）

21 ~ 22　以段落多少來表示的檔長度(低拜／高拜)CATALOG 目錄命令只可指到長度字之低拜部份（ 1 ~ 255 ），實際上，滿載時在此位置上可指達 65,535拜長。

圖 4.4 爲一典型的目錄段落構造。在此例中整個磁碟內共包含有 4 個檔，所以只需一個目錄段落就夠了。只有 4 個檔案進入點被使用其餘 3 個未被用而包含零。

軌道/段落表

每一個檔都有一個"軌道／段落表"段落。在軌道／段落表段落內此檔所包含之所有段落指標都照著順序排列著。在目錄段落內，檔進入點索引有一指標指向軌道／段落表，軌道／段落表再指向檔之各個段落。此觀念圖示於圖 4.5 。

軌道／段落表的格式如下所示。注意，即使一個檔小的端也需

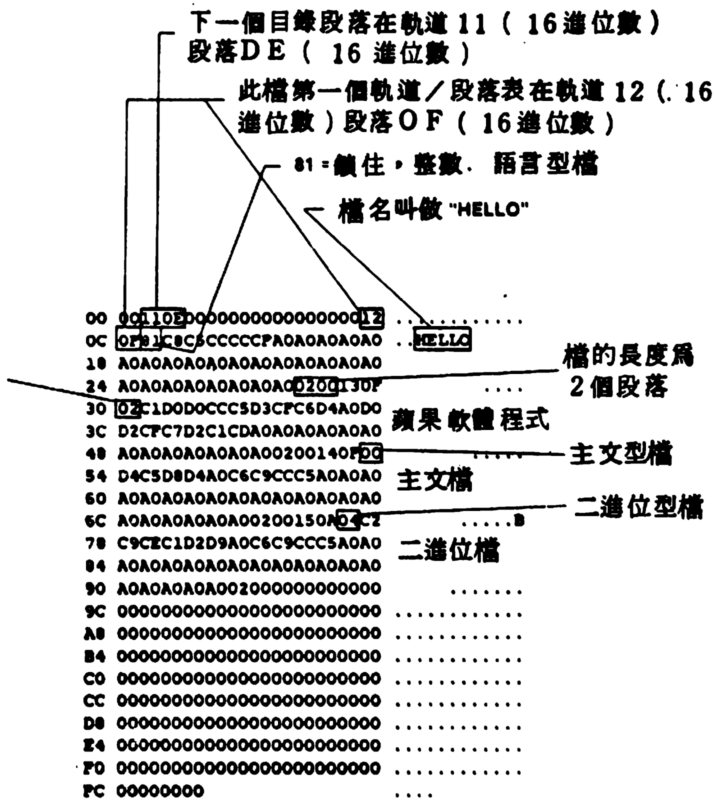


圖 4.4　目錄段落之例

要一個軌道／段落表段落及另一個段落來存放資料。所以一個非空的檔最少需用掉兩個段落。又，若是一個非常大的檔，其使用之資料段落超過 122 時，則需要有 2 個以上之軌道／段落表段落來裝下這些資料段落的指標。

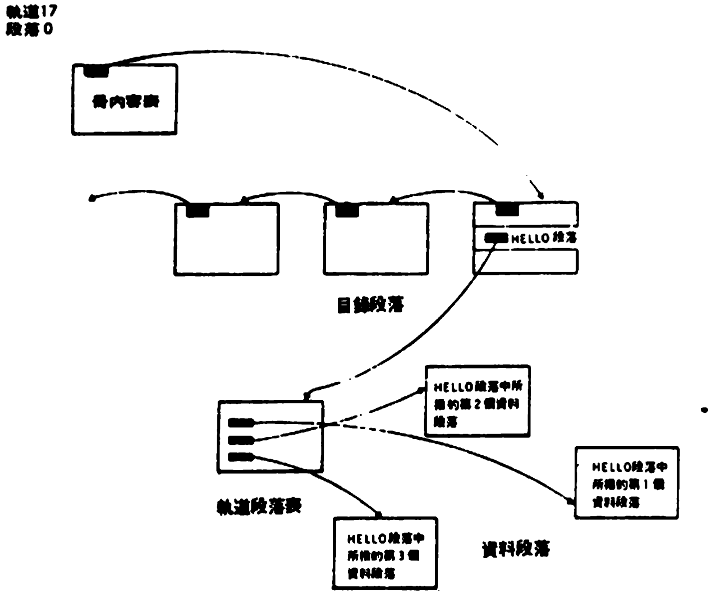


圖 4.5　DOS 找出一個檔資料所經過的路程

軌道/段落表格式

拜	內容
00	未使用
01	下一個軌道／段落表段落所在之軌道號數。若此檔不需兩個以上之軌道／段落表則此拜為 00 。
02	下一個軌道／段落表段落之段落號數(如果有下一個軌道／段落表的話)。
03 ～ 04	未使用

05 ~ 06	檔之第一個段落在此檔內之段落偏移量
07 ~ 0B	未使用
0C~0D	第一個資料段落所在的軌道及段落位置
0E~0F	第二個資料段落所在的軌道及段落位置
10 ~ FF	第三到第 120 個段落之軌道及段落位置（注意，若上述之軌道及段落位置爲0000 的話則表示此檔到此結束。）

圖 4.6 顯示一軌道／段落表。

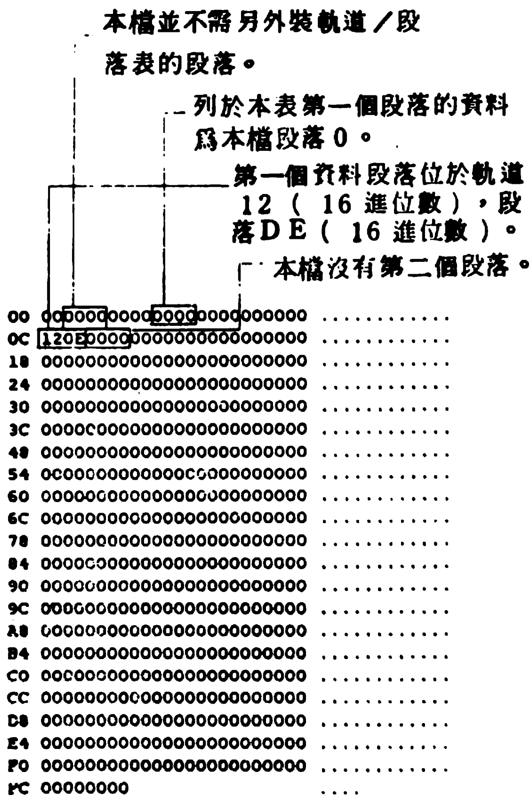


圖 4.6　軌道／段落表之例

一個次序排列檔在軌道及段落表內第一次碰到全部爲零的段落時結束，但一個隨機檔則不同，其軌道／段落表中會有間斷的空間（未對應到段落）存在。DOS 並非永遠都能處理這樣的隨機檔，例如，"查證"（VERIFY ）命令在軌道／段落表上一遇到空的段落指標就自動停止，因此不能用於隨機檔之結構上。（RANDOM FILE）

在圖 4.6 所舉的例中，檔只含有一個段落因爲其資料不大於256拜，把軌道／段落表段落一起算下去，該檔的長度就爲兩個段落，當我們使用"目錄"（CATALOG）命令時，此值就會用來顯示此檔之長度。

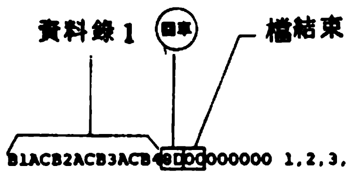
由軌道／段落表中之指標，我們可以找到第一個資料段落。檢查一下這些資料段落，不同類別的檔其間的差別將會很明顯。所有的檔（也許除了亂序檔外）都是由連續的資料所構成。如果資料過長，它們將會被分成 256 拜一段落以符合磁碟規格之要求。實際上這些段落並不需連續的放置在磁碟上，藉著軌道／段落表的指標，DOS 可以正確無誤的照著次序處理一個檔。程式員一點也不必擔心他的檔如何的被切成段落。

主文檔

主文檔的資料形式是最不複雜的資料結構。由是由許多資料錄（RECORD ）所組成，每個資料錄間由"回車"字母（carrage return ）（16 進位 8D，ASC Ⅱ ）所分隔。圖 4.7 顯示出此種結構及一個例子。通常，一個主文檔之終結是由所有資料成爲 00 或是由軌道／段落表中的段落指標爲零所獲知的。如果一個檔是亂序的構造，可能資料內就會有00出現或是在軌道／段落表中有零指標出



次序排列主文型檔




```
00 B1ACB2ACB3ACB40D00000000 1,2,3,4.....
0C 000000000000000000000000 ............
18 000000000000000000000000 ............
24 000000000000000000000000 ............
30 000000000000000000000000 ............
3C 000000000000000000000000 ............
48 000000000000000000000000 ............
54 000000000000000000000000 ............
60 000000000000000000000000 ............
6C 000000000000000000000000 ............
78 000000000000000000000000 ............
84 000000000000000000000000 ............
90 000000000000000000000000 ............
9C 000000000000000000000000 ............
A8 000000000000000000000000 ............
B4 000000000000000000000000 ............
C0 000000000000000000000000 ............
CC 000000000000000000000000 ............
D8 000000000000000000000000 ............
E4 000000000000000000000000 ............
F0 000000000000000000000000 ............
FC 00000000                 ....
```

例子：主文型檔的段落

圖 4.7 主文型檔的資料形式

現。這時唯一的辦法就是只好把軌道/段落表從頭到尾找一遍，找出最後一個非零的軌道/段落指標來。"回車"字母及16進位數字00在主文檔裏都具有特殊的意義，因此它們不能成爲資料的一部份。因爲這個原因及使資料能被BASIC 語言程式所用，資料只能包含可以印出來的ASCII 字母（數字、文字及特殊字母見蘋果二世DOS參考手册P.8）。這個限制使得處理主文檔時較慢且在使用磁碟空間時二進位型檔使用效率較差。因爲每一個10進位數必須佔掉一個拜的位置。

二進位型檔

圖4.8顯示二進位型檔之結構。首先寫入磁碟的是此檔在記憶體中之位址及其長度（共需要4個拜），位址及長度是以先低拜後高拜的格式表示，它們是由產生此檔的BSAVE命令裏面的鍵語A及L所產生（A代表Address ，L代表Length）。注意，DOS 在每一個檔後面多加了一個拜。但這並沒有什麼關係，因爲像BLOAD及BRUN這些命令將會先檢討檔有多長，而只處理相當於其長度之資料（當然這時檔之長度剛好是256拜時，就要浪費一個段落了）。DOS 有兩種方法把這些二進位程式映像放到記憶體不同的位址上去，第一種是當命令進入後，提供一位址鍵語，第二種爲改變磁碟檔最先頭兩個拜所含之位址資料。



二進位型檔

開始位址 = 03D0（16 進位數）

檔長度 = 0030（16 進位數）

從記憶體載入之二進位型映像

```
00 D00330004CBF9D4C849D4CFD P.O.L?.L..L]
0C AA4CB5B7AD0F9DAC0E9D60AD *LS7-..,..`-
18 C2AAACC1AA604C51A8EAEA4C B*,A*'LQ(JJL
24 59FABF9D384C58FF4C65FF4C YZ?.8LX_Le_L
30 65FF65FFBE00000000000000 e_e_>.......
3C 000000000000000000000000 ............
48 000000000000000000000000 ............
54 000000000000000000000000 ............
60 000000000000000000000000 ............
6C 000000000000000000000000 ............
78 000000000000000000000000 ............
84 000000000000000000000000 ............
90 000000000000000000000000 ............
9C 000000000000000000000000 ............
A8 000000000000000000000000 ............
B4 000000000000000000000000 ............
C0 000000000000000000000000 ............
CC 000000000000000000000000 ............
D8 000000000000000000000000 ............
E4 000000000000000000000000 ............
F0 000000000000000000000000 ............
FC 00000000                 ....
```

例子：二進位型檔之段落

圖 4.8 二進位型檔之資料形式

蘋果軟體及整數檔

一個以BASIC 程式語言所寫之程式，不管是蘋果軟體型檔或是整數型檔，是以一種和BSAVE 命令所做成的事相同的方式被存入磁碟當中。蘋果軟體型檔的格式圖示於圖 4.9 而整數BASIC語言型檔格式則圖示於圖 4.10。當"存入"（SAVE）命令輸入時，DOS 就會決定BASIC程式在記憶體所存的位置及其長度。因為BASIC 程式所在的位置BASIC 編譯器一定知道，所以在其檔中並不需像二進位型檔那樣在最前頭存一個位址。故現在所存的只是檔的長度及從記憶體抄過來的程式。注意，DOS 此時仍會多寫一個拜在檔裏，雖然在執行"載入"（LOAD）這命令時這個拜完全不被理會。程式之記憶體映像（亦即存於磁碟之資料）表示每一行的資料的都被轉換成一種叫"代號"（token）的內在格式，而一個程式映像被搬到記憶體後如何處理就不在本書討論範圍內了。

其它型檔(S,R,新A,新B型)

DOS 還有其它的檔型，在以前所說的索引進入點格式可以看出。到現在為止，還沒有任何DOS 指令可施行到這些檔型，所以其用途至今仍未確定。但其中R 檔的型曾在DOS TOOL-KIT 組合機中被用過，當做此組合機的輸出檔——一種可重定址的目標模組。這種檔型必須與特殊的二進位型檔配合使用。這種二進位型檔能夠包含一種機器語言映像，這種機器語言能由映像中之資料隨意定址。若要詳細知道這種檔之格式請參考DOS TOOL-

低拜/高拜格式	從記憶體載入之程式映像

長度

蘋果軟體型檔

```
10 PRINT   "[CTRL-D] OPEN TEXT FILE"
20 PRINT   "[CTRL-D] WRITE TEXT FILE"
30 PRINT   "1,2,3,4"
40 PRINT   "[CTRL-D] CLOSE TEXT FILE"
50  END
```

程式有5E(16進位數)個拜長

蘋果軟體程式

```
00 5E0018080A00BA22044F5045 ^.....:".OPE
0C 4E20544558542046494C4522 N TEXT FILE"
18 0030081400BA220457524954 .0...:".WRIT
24 4520544558542046494C4522 E TEXT FILE"
30 003F081E00BA22312C322C33 .?...:"1,2,3
3C 2C342200570828000BA220443 ,4".W.(.:".C
48 4C4F5345205445585420 4649 LOSE TEXT FI
54 4C4522005D08320080000000 LE".].2.....
60 640000000000000000000000 d...........
6C 000000000000000000000000 ............
78 000000000000000000000000 ............
84 000000000000000000000000 ............
90 000000000000000000000000 ............
9C 000000000000000000000000 ............
A8 000000000000000000000000 ............
B4 000000000000000000000000 ............
C0 000000000000000000000000 ............
CC 000000000000000000000000 ............
D8 000000000000000000000000 ............
E4 000000000000000000000000 ............
F0 000000000000000000000000 ............
FC 00000000                 ....
```

Example Applesoft File Sector

圖 4.9

低拜／高拜格式	從記憶體搬入之程式映像
長度	

整數型檔

10 END

程式共 5 個拜長

此條指令長度（5 個拜）

指令編號（OAH = 10D）

"代號" { 51 = END; 01 = 本條指令結束 }

```
00 05000050A0051010500000000 .....Q......
0C 0000000000000000000000000 ............
18 0000000000000000000000000 ............
30 0000000000000000000000000 ............
3C 0000000000000000000000000 ............
48 0000000000000000000000000 ............
54 0000000000000000000000000 ............
60 0000000000000000000000000 ............
6C 0000000000000000000000000 ............
78 0000000000000000000000000 ............
84 0000000000000000000000000 ............
90 0000000000000000000000000 ............
9C 0000000000000000000000000 ............
A8 0000000000000000000000000 ............
B4 0000000000000000000000000 ............
C0 0000000000000000000000000 ............
CC 0000000000000000000000000 ............
D8 0000000000000000000000000 ............
E4 0000000000000000000000000 ............
F0 0000000000000000000000000 ............
FC 00000000 ....
```

例子：整數型檔段落

圖 4.10 整數 BASIC 型檔資料形式

KIT 程式及相關文件。

緊急修護

不時的，存在磁碟上的資料會受到傷害甚至損失掉。發生時總會有點徵兆。從一些溫和的副作用，如磁碟因此而不能啓動，到大麻煩，例如目錄內資料的錯誤。一個精明的蘋果二世使用者能夠修補大部份磁碟的缺失，如果它對前所述之磁碟格式有很好的了解及會使用一些程式工具的話。

第一個問題將是"這缺失是如何發生的？"，缺失發生的最普通的原因是磁碟受到損傷。經常的，磁碟也會藉著軟體警告已受到傷害了。軟體警告是在作輸出入之發生，它的產生是隨機的（ ron-dom）。你可能在作登錄命令時得到一個軟體警告，但再重作一次這警告又消失了。這時，最好是再找一塊新磁碟把舊磁碟的資料抄進去，舊磁碟可扔掉或留作備用。



如果有一份備用的磁碟的話，緊急修護就會變得較簡單

另外一種損傷磁碟的原因是程式員在使用磁碟時不小心碰到"重置"（RESET）鍵。當磁碟正被寫入資料時若有重置信號進來時就會使得磁碟受到傷害。此外，像在寫入資料時停電，或磁碟機、控制板及電線硬體發生問題都可使磁碟遭受到傷害。

如果損傷的磁碟能夠做"目錄"動作時，那麼修護就容易多了。如果在前三個軌道所存的DOS 程式映像遭到損傷的話，我們可以執行"主磁碟創建"（MASTER CREATE）程式來修正錯誤，或者乾脆把所有的檔都抄到另一磁碟。若是執行"查證"（Verify）命令時，只有一個檔發生輸出入錯誤，那麼在抄入新磁碟時只要把損害的那個段落跳過即可（你可寫一個組合程式叫出DOS 內的RWTS，或者寫個BASIC程式(如果此檔是個主文檔)來達到此目的）。事實上，如果錯誤僅是偵錯區不符，你仍可從這"損傷"的段落裏讀出大部份仍可用的資料。

一個輸出入錯誤通常意味著發生了一二件事件。可能是偵錯區不符而警告使用者此段落所含資料"不佳"，或者，資料嚴重毀損到已不存在於此磁碟上。如果是後者，那麼此磁碟（最少是此軌道）內資料必須重新排列。雖然DOS 能對一個軌道稍作修補，通常較簡單的作法是把所有可讀的段落從損害的磁碟存到另一個格式化的磁碟，然後在此新磁碟上重建那些遺失的資料。

許多設備都有使用者可讀出或顯示出段落內容的設施。其中有些還允許使用者修改段落後再重新寫入磁碟（舊的新的均可）中。這種設施的一個簡單例子見附錄A。在那裏，ZAP 程式可以把任何軌道或段落（如果是未保護檔的話）讀入記憶體中讓使用者觀看或修改，然後，隨著使用者意願，重新寫入磁碟中。在學習磁碟格式及修護受損的磁碟時，這個程式相當重要。

使用 ZAP 程式，在一個檔中"不佳"的段落能夠被找出。辨

法是讀出檔之軌道段落表段落之內容，如果這"不佳"的段落是一個目錄段落，那麼將有 7 個檔的指標會遺失掉。如果這種情形發生的話，就要開始檢查整個磁碟，把其它那些"佳"的段落對應到軌道段落表上，最後剩下來在軌道段落表上找不到對應的代表其指它的資料已損壞。新檔的索引進入點可以塡入這些損害資料的地方。這些步驟，即使在熟知DOS 磁碟格式的情況下，也須要花好幾個小時。作這些步驟是在已沒有其他更好的辦法來恢復資料的情形下爲之。當然，最好的維護辦法是對那些重要的檔週期的在另一片磁碟上再建一個備用檔，這樣維護就變得很簡單了。上述步驟的詳細資料請參考附錄A。

另有一種較不明顯的磁碟傷害，但是也很令人困擾，就是未使用段落被遺失了。當一個檔被開啓時，DOS 同時配置了整個軌道的段落來處理這件事。不小心碰到重置鍵（RESET）能夠使這些段落在册內容表（VTOC）內都被註上"使用中"的記號，但這些段落通通尙未被排列到軌道段落表上去。此時，這些失去的段落就不能以一般的方法來恢復。即使把此檔刪除，由於這些段落亦不在該檔的軌道段落表上出現，仍然無補於事。此種情形造成系統提前通知你磁碟已滿，而實際上磁碟仍未滿之情況。要找出這些丟掉的段落，我們必須把軌道段落表所指向的段落與册內容表（VTOC）內之比次對映資料慢慢來比對，以找出錯誤來。當然也有現成的程式來做此工作，但解決此問題之最佳方法是把此磁碟所包含的所有檔案都複置到另一磁碟上去。（注意，此時需用FID 命令而非COPY命令），因此COPY命令把整張磁碟，不論好壞，都一古腦複製過去了。

如果一個檔被刪除了，通常可以重新恢復，只要被刪除以後，沒有新檔蓋掉相同的位置就可以。如果在刪除命令後DOS 又產生

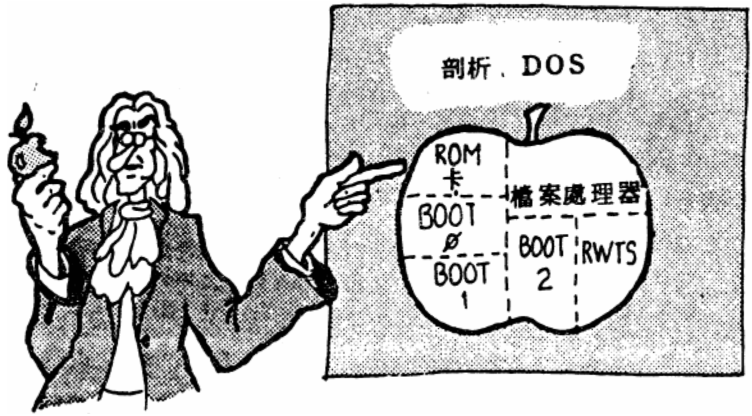
新的檔進來，那麼原來檔段落的一部或全部也許會被用掉。此時，目錄會很快的被改變，即軌道段落表的軌道編號從檔描述進入點的第20拜移到第 0 拜。最好的辦法是把想刪除的檔在另一片磁碟複製後再把原有磁碟的檔刪除。這樣，冊內容表的自由空間比次對應才能放心的被更新。

第五章 DOS的結構

DOS記憶體之使用

DOS 是以組合語言寫成的程式，當使用者BOOT磁碟時，它便放入RAM 記憶體中。BOOT主磁碟時（MASTER DISK）時，DOS 映像依所使用機器的大小不同而放入其RAM 的末段位址上。如此一來，DOS 就必須使BASIC 了解到它可用到的RAM少於系統原有確實的數量。例如，在 48 K APPLE Ⅱ之中作用DOS 時，BASIC 只有 38K RAM可資利用。DOS 放入記憶體後，以調整HIMEM的方法防止BASIC 使用到DOS 所佔的記憶體。BOOT副磁碟時（SLAVE DISK），DOS放入起始設定此磁碟時（INITIALIZED ）它所在的位置。若副磁碟建於 16K A APPLE 上，DOS 便放入第 6 至16K RAM範圍中。如此一來，APPLE 實際上只剩6K RAM 可供應用。若副磁碟建於 48K APPLE 上，它便無法在小於48K系統上BOOT。因爲副磁碟上DOS 所佔的RAM 位置在較小系統上不存在。

圖 5.1 顯示 48K APPLE Ⅱ DOS 的記憶位置安排情形。其中含有 4 個主要區間，爲首的 1.75 K爲檔案緩衝器（FILE



BUFFERS ）。由於自然值（DEFAULT ）定為MAXFILES 3，此處含有3組檔案緩衝器。每個緩衝器對應一個開檔案（OPEN FILE），佔有595個拜（BYTES）。DOS利用檔案緩衝器處理LOAD及SAVE FILE等工作。若將MAXFILES數目改掉，則檔案緩衝器所佔的空間發生變化，影響HIMEM指標，使其根據緩衝器空間之變化（變小或變大）往上或往下移動。

檔案緩衝器上方3.5 K位置存DOS 主要常式（ROUTINES）可執行的DOS 機器語言碼起始於此。主要常式功能包括起始設定DOS，對BASIC界面，解釋命令（COMMENDS），及處理檔案緩衝器事務。所有磁碟的功能函數（FUNCTION）都經由呼叫檔案處理器而傳出執行。

佔2.8 K的檔案處理器集合了許多負責執行各種存取磁碟所須的功能。這些功能包括OPEN，CLOSE，READ，WRITE，POSITION，DELETE，CATALOG，LOCK，UNLOCK，RENAME，INIT，及VERIFY。檔案處理器雖是在DOS之中的副程式，使用者的組合語言程式仍可透過界面呼叫它。此界面概括為一群在RAM第三頁的向量指標（VECTORS），下一章將會詳細說明它

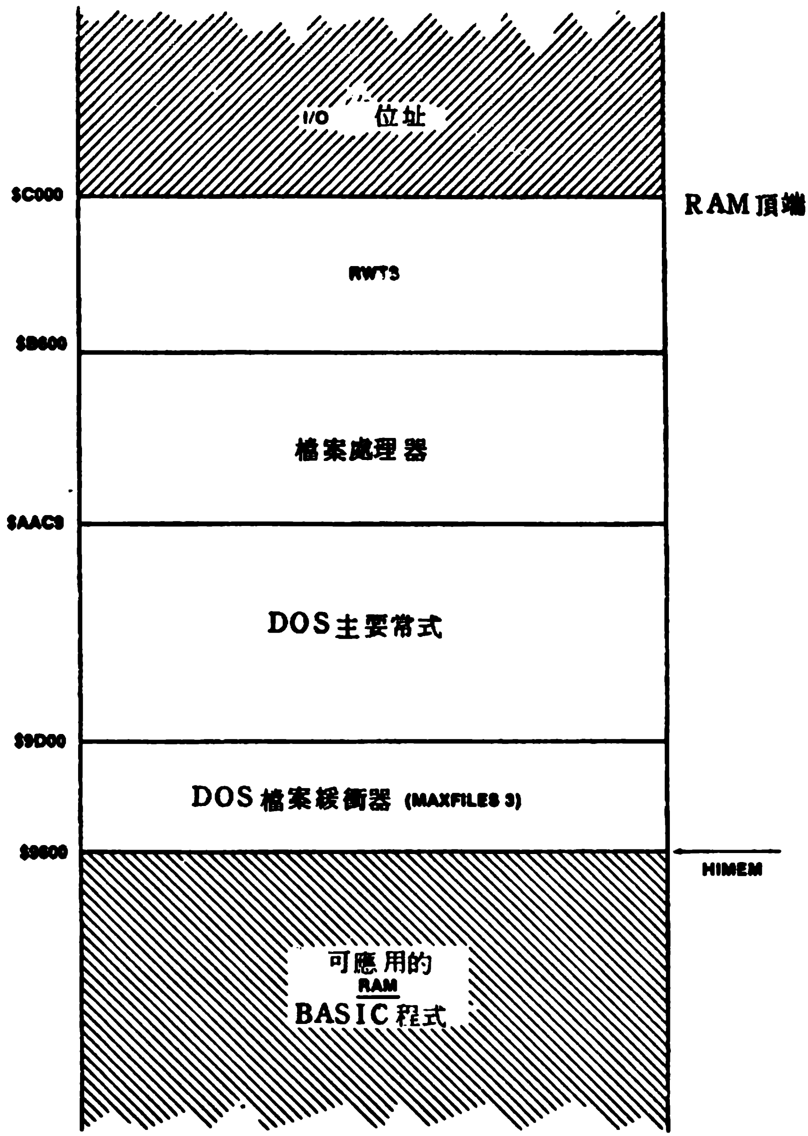
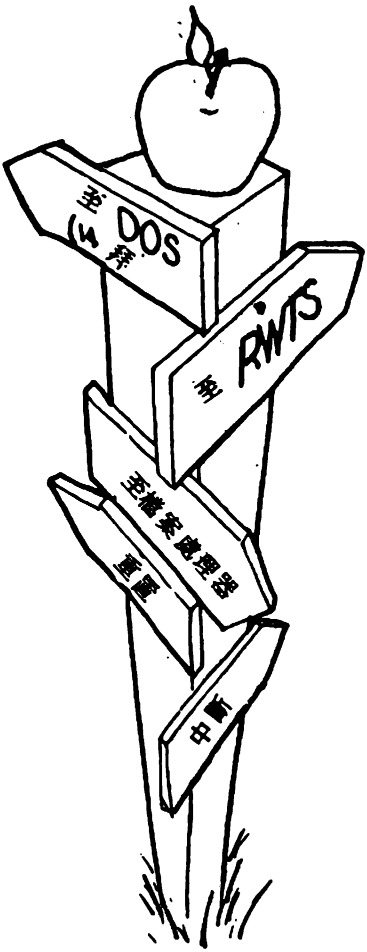


圖 5.1 DOS 記憶體的使用(48K APPLE)

們。

DOS 的最後2.5K位置是R/W,T/S(讀/寫,軌道/段落)(RWTS)包裝組。RWTS是檔案處理器的次級規約,實際上它只是個檔案處理器所呼叫的副程式。檔案處理器處理檔案,而RWTS處理磁碟上軌道及段落的問題。呼叫RWTS的典型例子如將17軌道第0段落中之資料讀出,或將記憶體中256個拜資料寫入第5軌道第E段落中。DOS也具有外部界面以讓使用者的組合語言程式可得用得到RWTS,下一章亦將詳細說明它們。



DOS 向量指標

記憶體第三頁的DOS向量指標

有大約10K的高位址RAM被DOS 佔用以外,在低位址記憶體的第3頁處($300到$3FF),DOS亦保有空間以儲存一群向量指標。經由某固定位置(如:$3D0)可利用此向量指標進入DOS 常式集的特定點。因DOS 放入記憶體的位置隨著機器大小及使用的磁碟(主磁碟或副磁碟)而不同,在DOS 之中

可由外界呼叫的副程式其位址並不固定。若將副程式位址存於固定位置的向量指標，此困難便迎刃而解。至於不同版本的DOS ，其副程式位置亦可能不同。此時第三頁的向量指標表在定位這些副程式方面亦大有貢獻。DOS 早期版本在BOOT過程存入BOOT1 程式時使用到$200至$3CF 位置，但在DOS 3.3中，此位置用作資料緩衝器及磁碟碼轉換表。也許，這項更改的目的在於可供應較多的記憶體給予第一階段BOOT載入程式使用。向量指標表本身則自$3D0開始。

DOS向量指標表($3D0-$3FF)

位址	使用說明
3D0	存JMP 指令，將程式跳越至DOS 的暖啓動常式(WARMSTART ROUTINE)。這常式讓使用者重入DOS，但並不取消正在執行的BASIC 程式，也不重置MAXFILES或其它環繞DOS 的變數。
3D3	存JMP 指令，將程式跳越至DOS 的冷啓動常式(COLDSTART ROUTINE)，以重新起始設定DOS。與REBOOT DOS的作用一樣，它清除了正在執行的BASIC檔案並重置HIMEM。
3D6	存入JMP 指令，將程式跳越至DOS 的檔案處理器，讓使用者的組合語言程式可呼叫它。
3D9	存JMP指令，將程式跳越至DOS 的RWTS常式，讓使用者的組合語言程式可呼叫它。
3DC	呼叫一短的副常式以存放檔案處理器所須的輸入參數表，讓使用者在呼叫檔案處理器之前可先設定適當的輸入參數
3E3	呼叫一短的副常式以存放RWTS 所須的輸入參數表，讓

使用者在呼叫RWTS之前可先設定適當的輸入參數。

3EA　存JMP 指令，將程式跳越至一副程式，以便重新連接(RECONECT) DOS 截斷器(INTERCEPT)到鍵盤及銀幕資料串(DATA STREAM)上。

3EF　存JMP 指令，將程式跳越至處理機器語言指令"BRK"的常式。內含顯示暫存器內容的副程式(在監督程式ROM 之中)的位址。本向量指標僅由自動啓動ROM (AUTOSTART ROM)給定。

3F2　存有供自動啓動ROM 使用的重置(RESET)常式的位址(低拜/高拜)。常態下，存著DOS 重新啓動(RESTART)的位址，但使用者若欲自行處理RESET 動作時，可更改它。

3F4　開機拜(POWER-ON BYTE)，內存與RESET位址互補的值$A5，用以判別是機關動作或是RESET按鍵動作。開機時。自動啓動ROM 不執行在$3F2 的功能(因尚未起始設定)而執行BOOT磁碟過程。爲了避免使用者於更改$3F2 內容以處理自己的RESET 時，發生誤認爲開機事件，使用者應將$3F2 內的值與$A5 作EOR(EXCLUSIVE OR)運算再存回$3F4 中。

3F5　將程式跳越至一機器語言常式。在執行APPLESOFT，當使用到"&"特徵符號(FEATURE)時呼叫此常式。

3F8　將程式跳越至一機器語言常式。當"CONTROL-Y"鍵入時呼叫此常式。

3FB　將程式跳越至一機器語言常式。當不可罩(NONMASK-ABLE)中斷發生時呼叫此常式。

3FE　存有一常式的位址(低拜/高拜)，當可罩(MASKABLE

）中斷發生時呼叫此處的常式。

BOOT磁碟到記憶體的過程

APPLE 開機時，記憶體中並無任何程式。BOOT磁碟以獲取

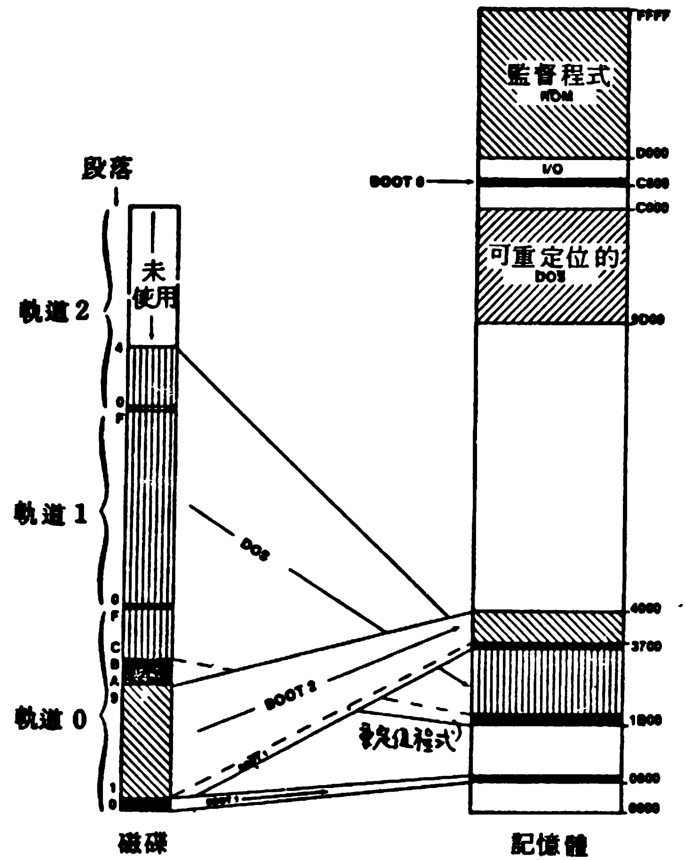
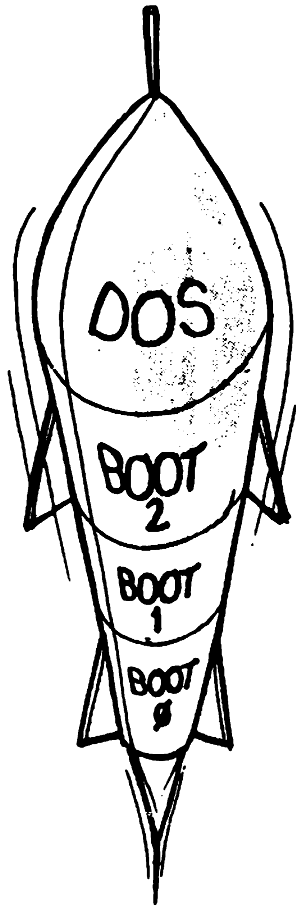


圖5.2　BOOT過程

DOS。BOOT就是將DOS 以靴式載入RAM 的過程。BOOT過程包含一系列步驟，把DOS 程式的片段自小至大依次地載入記憶體中，直到所有程式完全進入記憶體中並可用來執行為止。BOOT在此DOS 中分為四個階段。這四個階段在磁碟及記憶體上的對應位置可由圖5.2得之。以下詳述BOOT過程。

第一階段BOOT（稱為BOOT0）執行儲存於磁碟控制卡上ROM 中的程式。使用者鍵入PR#6，C600 G，或6↑P（↑表CONTROL），CPU控制權便交由在第六接線槽上磁碟控制卡的ROM。此ROM 中含有256個拜的機器語言程式。當執行時，它將磁碟探臂拉回第零軌道上以重校準探臂位置（可聽到咔啦的聲音）。然後將第零軌道第零段落的內容自RAM 的$800 處讀入RAM（早期DOS版本自$300 處讀入）。讀完後第一階段BOOT便令程式跳越至$800 位置進行第二階段BOOT的工作（稱為BOOT1）。

BOOT1 也有256 個拜。它利用了部份BOOT0ROM中的副程式將第零軌道上另外9個段落（第1段至第9段）內容讀入RAM 中。這些段落內含次階段BOOT過程（稱為BOOT



BOOT 過程

2)。根據所使用磁碟種類的不同(主磁碟或副磁碟),BOOT2載入記憶體中不同的位置。若使用副磁碟,BOOT2載入DOS所在記憶體最後 9 頁(每頁 256 個拜)位置上,此處即副磁碟最初被起始設定的地方。也就是,若副磁碟原建於 32KDOS ,BOOT2 就會載入 $ 7700 至$ 8000 RAM 中。若使用主磁碟,BOOT2 載入 $ 3700 至 $ 4000位址RAM中與副磁碟在16K 系統情況下相同。於載入BOOT2 的記憶體下方,再度載入 BOOT1(主磁碟情形下是在 $ 3600),以便當起始設定一新磁碟時可直接將 BOOT1 複製到磁碟的第零軌道第零段上。當BOOT1 執行完且載入BOOT2 之後,程式便跳越至BOOT2 以進行下階段BOOT工作。

BOOT2 包括載入程式(LOADER)的"主程式"及RWTS副程式。到此爲止磁碟探臂因僅在第零軌道上讀取而未移動過。此後探臂須移至其他軌道上以取得更多段落資料以載入記憶體。處理複雜的磁碟存取工作,BOOT2 載入程式利用RWTS副程式專責控制磁碟探臂的移動並且將BOOT最後一部分(也就是DOS 本身)所在的段落內容載入記憶體中。BOOT2 將探臂定於磁碟第二軌道第四段落上,並將其內容讀入記憶體中BOOT1 所在位置的下方(使用主磁碟時,位址爲 $3500)。然後BOOT2 連續讀入26個段落資料(除了最後一段外,每段有 256 個拜)。使用主磁碟時,最後並將第零軌道第A段落內容自$1B00讀入。這27個段落內容就是DOS 主常式群以及檔案處理器。經過這些載入過程後所有的DOS 均已在記憶體之中了。此時副磁碟的BOOT過程已大功告成,程式並跳越至DOS 冷啓動程式(COLDSTART)的位址去。在使用主磁碟情形下,此時僅 16K APPLEII才能使用在記憶體中DOS 的映像,若系統大於 16K,DOS必須再移到最高的記憶位置上去。因此程式先跳越到 $1B03 重定位程式位址上。重定位程式

長 512個拜，它已事先存於DOS 映像的最低兩頁中（使用副磁碟時，這兩頁填入"零"）。重定位程式有系統地做儲存及載入的動作於較高頁RAM ，以決定機器大小直到找到最高頁爲止。之後它便將DOS 自$1D00 往上移，移到規定的位置上去（48K系統中爲$9D00），並利用表格修正程式中機器碼使程式在新位置上能夠正常工作。然後重定位程式便跳越到高位置DOS 上並將低位置DOS 捨棄掉。

冷啓動程式起始設定DOS ，爲檔案處理器準備空間，預定HIMEM，建立第三頁向量指標表，並執行HELLO程式。DOS BOOT過程到此全部完畢。

早期DOS 版本的BOOT過程稍嫌繁複。它們的BOOT1 載入$300，及BOOT2 載入$3600與DOS3.3相同。但它們的27個段落並不全部載入記憶體。副磁碟情況下，只有25個段落載入。且若所使用的磁碟爲13個段落式的（每一軌道），其DOS 映像的終止位置依副磁碟或主磁碟不同分別在第二軌道第8或第A段落。並且其BOOT1 在磁碟中的分塊（NIBBILIZATION ）型式與其他段落不同，單純地將它載入記憶體中並不能用來執行。

各階段BOOT過程將於第8章DOS 程式邏輯中再詳予說明。

第六章 以組合語言使用DOS的方法

注意：

本章用來使資深的組合語言程式設計師能夠不經由BASIC PRINT 敍述而存取磁碟。這裡面所包含的題材可能令未學過組合語言的使用者無法了解。

一、直接使用磁碟機

不經過DOS 而以組合語言直接存取磁碟的方式常是有用且必要的。如此做要靠與硬體直接界面的16個位址門住觸發器。其中8個雙拜觸發器實際上各拉動著一條TTL 控制線。自使用者所寫的作業系統以至與DOS 不相關的公用程式（UTILITY PROGRAM）都可做直接存取磁碟的應用。圖 6.1 顯示裝置位址的安排法。



本章供程式設計師選讀

位址	標記	說明
$C080	PHASEOFF	步進馬達0相OFF（斷路）
$C081	PHASEON	步進馬達0相ON （通路）
$C082	PHASE1OFF	步進馬達1相OFF
$C083	PHASE1ON	步進馬達1相ON
$C084	PHASE2OFF	步進馬達2相OFF
$C085	PHASE2ON	步進馬達2相ON
$C086	PHASE3OFF	步進馬達3相OFF
$C087	PHASE3ON	步進馬達3相ON
$C088	MOTOROFF	關閉馬達
$C089	MOTORON	開動馬達
$C08A	DRVOEN	第1號磁碟推動器作用

$C08B	DRV1EN	第2號磁碟推動器作用
$C08C	Q6L	觸動"資料閂住器"使其用於輸入輸出
$C08D	Q6H	載入"資料閂住器"
$C08E	Q7L	準備輸入閂住器
$C08F	Q7H	準備輸出閂住器

Q7L及Q6L=讀

Q7L及Q6H=感測"寫入保護"之狀態

Q7H及Q6L=寫

Q7H及Q6H=載入"寫入閂住器"

圖 6.1 DOS硬體位址之配置

這些位址與電路板所在槽路有關，其偏移量爲槽路號數乘16。在16進位制中可視爲這些基底位址加上偏移量 $S0（S表槽路號數）。若令在第6槽路上的第1號磁碟推動器作用，應將 $C08A 加 $60 成爲 $C0EA。但使用與槽路無關的寫法較方便，一般使用 $C08A,X（X暫存器內存 $60）。

只要上述位址可取到，使用何種方式的 6502 指令並無差別。但在讀／寫拜時，須謹慎地將資料放入正確的暫存器上。下列每個指令都可用來使第1號推動器作用（位於第6槽上）。

```
LDA   $C0EA
BIT   $C08A,X( X暫存器內容爲$60 )
CMP   $C08A,X( X暫存器內容爲$60 )
```

以下舉出幾個例子，以示範如何使用所安排之裝置位址。它們

都設於第 6 槽上，X暫存器內容皆爲＄60。附錄A有更多例子供參考。

1.步進馬達各相的開與關

基本上步進馬達的動作，其四個相中的每相都須開而復關。四個相以下數的順序開關之可使探臂向磁碟中心移動。以往上數的順序開關之使探臂向外移動。存取這些位置之間的時序必須非常準確並恰到好處。這留給讀者當作一個很好的練習。在此我們建議使用RWTS中的 SEEK 命令移動探臂較爲方便。參看下述RWTS的使用說明。

2.馬達的開與關

```
LDA  $C089,X   關閉馬達
LDA  $C089,X   開動馬達
```

註：必須充分提供給馬達啓動到達定速之時所須的延遲時間。此時間可爲1秒，但DOS 會自動檢查〝讀入門住器〞判別資料是否開始變動，以此法減少延遲時間。

3.使第 1 或第 2 推動器作用

```
LDA  $C08A,X    第1 推動器作用
LDA  $C08B,X    第 2 推動器作用
```

4.讀入一拜

```
READ    LDA $C08C,X
        BPL READ
```

註：$C08E,X須用在前，以確定處於輸入模式。須利用環路（LOOP）以檢定累積器（ACCUMULATOR）是否內存正確值。當資料閂住器內未存有正確資料時，高比次會是0。

5.感測"寫入保護"之狀態

```
LDA $C08D,X
LDA $C08E,X     感測"寫入保護"之狀態
BMI ERROR       若高比次爲1（SET），表示已獲保
                護
```

6.載入"寫入閂住器"及寫入一拜

```
LDA DATA
STA $C08D,X   載入"寫入閂住器"
ORA $C08C,X   寫入此拜於磁碟
```

註：$C08F,X須用於前，以確定處於寫入模式。並須在100微秒之後才可開始寫入。

因硬體的限制，資料拜必須使用花掉32週（CYCLES）的環路以寫入磁碟。以下爲資料載入堆積器（ACCUMULATR）然後寫入磁碟的例子。由於時序要求十分嚴格，依取得資料方法的不同必須使用不同的常式，且若非經謹愼的調整過，這些程式碼不得跨越兩個記憶"頁"的邊界。

```
          LDA  #$D5        ( 3週 )
          JSR  WRITE9      ( 6 )
          LDA  #$AA        ( 3 )
          JSR  WRITE9      ( 6 )
            ⋮
WRITE9    CLC              ( 2 )
WRITE7    PHA              ( 3 )
          PLA              ( 4 )
WRITE     STA  $C08D,X     ( 5 )
          ORA  $C08C,X     ( 4 )
          RTS              ( 6 )
```

二、呼叫RWTS程式

每個DOS 版本都具有RWTS程式。它是個副程式集合，約佔據DOS 上方三分之一的位置。RWTS具有標準界面及完整的文件敍述供非DOS 的程式呼叫它。

有兩個副程式必須被呼叫並執行之。

JSK $3E3——呼叫此程式時，DOS 的輸出入控制塊(INPUT/OUTPUT CONTROL BLOCK——IOB)的位址載入Y及A暫存器。其中Y暫存器存位址的低拜，A暫存器存位址的高拜。此副程式用來設定IOB 位置，其結果(Y及A暫存器之內容)可存於記憶體第零頁的某位置中供使用者輸入IOB 之適當值，並可於呼叫RWTS後由此取得RWTS的輸出值。使用者亦可建立自己的IOB ，在呼叫RWTS 之前改變Y及A暫存器內容使其指

向自己的IOB。

JSR $3D9——RWTS 常式主進入點。呼叫此處之前，須事先將敍述工作方式的IOB位址放入Y及A暫存器中。IOB內含下表中定義的適當資料（其位移量爲16進位制數值）。

輸入/輸出控制塊(BLOCK)———一般格式

拜	說　明
00	表格型式。須爲$01。
01	槽路號數乘16（S0：S=槽路號數。例：$60）。
02	推動器號數（$01或$02）。
03	册號數（$00表任意册）。
04	軌道號數（$00到$22）。
05	段落號數（$00到$0F）。
06/07	裝置特性表位址（低拜/高拜）。
08/09	用於讀/寫的256拜緩衝器位址。
0A	未用。
0B	部分段落上的拜數量（$00表256拜）。
0C	命令碼$00＝尋找（SEEK）。 $01＝讀（READ）。 $02＝寫（WRITE）。 $04＝格式（FORMAT）。
0D	返回碼——若返回碼不爲$00，中央處理器系的CARRY旗號設爲1。 $00＝沒有錯誤。 $08＝在起始設定時發生錯誤。 $10＝寫入保護錯誤。

$20＝册號數不符錯誤。

$40＝推動器錯誤。

$80＝讀出錯讀（已作廢）

0E	前一次存取的册號數	（必須被起始設定）
0F	前一次存取的槽路號數乘16	（必須被起始設定）
10	前一次存取的推動器號數	（必須被起始設定）

裝置特性表(DCT—DEVICE CHARACTERISTIC TABLE)

拜	說明
00	裝置型式（DISK Ⅱ應爲$00）
01	每軌之相數（PHASES）（DISK Ⅱ應爲$01）
02/03	馬達的時間計數（DISK Ⅱ應爲$EFD8）

註：RWTS 使用了第零頁的$48位置，APPLE監督程式亦用此位置保存P暫存器的內容。每當呼叫RWTS之後必須將$48位置設爲$00。

三、呼叫型式下的RWTS IOB內容

尋找（SEEK） 將磁碟探臂移動到所須要的軌道上

輸入：拜 00 —— 表格型式（$01）。

01 —— 槽路號數乘16（S0：S=槽路號數）

02 —— 推動器號數（$01或$02）

04 —— 軌道號數（$00至$22）

06/07 — DCT位址

0C	SEEK的命令碼（ $00 ）
0F	前一次存取的槽路號碼乘 16 。
10	前一次存取的推動器號數。
輸出：拜 0D	返回碼
0F	現在的槽路號數乘 16 。
10	現在的推動器號數。

讀（READ）	讀一段落進入指定的緩衝器
輸入：拜 00	表格型式（ $01 ）
01	槽路號數乘 16（S0：S＝槽路號數）
02	推動器號數（ $01或$02 ）
03	册號數（ $00 表任意册）
04	軌道號數（ $00 到$22 ）
05	段落號數（ $00 到$0F ）
06/07	DCT位址。
08/09	使用者的256 拜資料緩衝器位址
0B	每段的拜數量（ $00 ）
0C	READ 的命令碼（ $01 ）
0E	前一次存取的册號數
0F	前一次存取的槽路號數乘 16
10	前一次存取的推動器號數
輸出：拜 0D	返回碼
0E	現在的册號數
0F	現在的槽路號數乘 16
10	現在的推動器號數。

寫（WRITE） 寫一指定緩衝器內容進入磁碟一段落

輸入：拜	00	表格型式（$01）
	01	槽路號數乘16（S0：S=槽路號數）
	02	推動器號數（$01或$02）
	03	册號數（$00表任意册）
	04	軌道號數（$00到$22）
	05	段落號數（$00到$0F）
	06/07	DCT位址
	08/09	使用者的256拜資料緩衝器位址
	0B	每段的拜數量（$00）
	0C	WRITE的命令碼
	0E	前一次存取的册號數
	0F	前一次存取的槽路號數乘16
	10	前一次存取的推動器號數
輸出：拜	00	返回碼
	0E	現在的册號數
	0F	現在的槽路號數乘16
	10	現在的推動器號數

格式（FORMAT） 起始設定磁碟（但不將DOS 放入磁碟，建VTOC/CATALOG，或存入HELLO程式）

輸入：拜	00	表格型式（$01）
	01	槽路號數乘16（SO：S=槽路號數）
	02	推動器號數（$01或$02）
	03	册號數（$00 使自然值（DEFAULT）定為 254）

，前述的參數表須先準備妥當（此表內容隨呼叫型式定之），且X暫存器亦須預設為零或非零。

X＝0 ——若找不到檔案，則爲它分配位置（ALLOCATE IT）。

X≠0 ——若找不到檔案，不爲它分配位置。

建立新檔案而呼叫OPEN時，X設爲0，其他型式的呼叫X均設爲非零。

程式設計師須在記憶體中分配三個緩衝器供檔案處理器使用。這三個共佔557拜的緩衝器在每次相關的檔案使用到時，就必須給予檔案處理器利用。每個開檔案（OPEN FILE）都須保留在某部分緩衝器上。DOS將這個緩衝器保留在高位址記憶體上（如前章所述）。謹慎處理之下這些緩衝器可"借"自DOS所佔的區域，下一章會說明這個方法。

FIG. 6.2表示出在不同型式的呼叫下，檔案處理器所須的輸入參數。檔案隨理器參數表的格式如下：

檔案處理器參數表——一般格式

拜	說 明
00	呼叫型式：01＝開檔（OPEN）
	02＝閉檔（CLOSE）
	03＝讀（READ）
	04＝寫（WRITE）
	05＝除檔（DELETE）
	06＝目錄（CATALOG）
	07＝鎖制（LOCK）

	06/07	DCT位址
	0C	FORMAT的命令碼
	0E	前一次存取的冊號數
	0F	前一次存取的槽路號數乘16
	10	前一次存取的推動器號數
輸出：拜	0D	返回碼
	0E	現在的冊號數
	0F	現在的槽路號數乘16
	10	現在的推動器號數

四、呼叫DOS檔案處理器

每一個DOS 版本都含有檔案處理器。它是個副程式集，約佔據DOS 中央三分之一的位置。這些常式都具有界面以供非DOS的程式呼叫之。 APPLE 並未發表此界面的定義（其他公司亦不發表類似技術），但因常式經由向量指標呼叫之，且傳送的參考表格式所有DOS 版本中都未改變過，這些常式的可靠性卻是非常高。況且，最新的FID 公用程式（UTILITY）並使用了這些常式處理磁碟的檔案。

有兩個副程式須被呼叫並執行之。

JSR $3DC——呼叫此副程式時，檔案處理器參數表的位址載入Y及A暫存器，其中Y暫存器存位址的低拜，A暫存器存位址的高拜。此副程式至少須呼叫一次以訂定參數表的位置。其位址值（在Y及A暫存器中）存入記憶體零頁的某位置以供程式設計師輸入參數表適當值，並於呼叫檔案處理器之後取得其輸出值。

JSR $3D6——檔案處理器的主要進入點，呼叫此程式之前

	08＝解制（UNLOCK） 09＝重定名（RENAME） 0A＝定位（POSITION） 0B＝起始設定（INIT） 0C＝查證（VERIFY）
01	READ或WRITE之次呼叫： 00＝不用作（不理會呼叫） 01＝讀或寫一拜 02＝讀或寫一群拜 03＝定位（POSITION）後讀或寫一拜 04＝定爲後讀或寫一群拜
02/09	各呼叫型式所使用的參數。參看以下"呼叫型式的檔案處理器參數表"（FILE MANAGER PARAMETER LIST BY CALL TYPE）。
0A	返回碼（註：並非所有的返回碼都可能出現在任何型式的呼叫）。若返回碼不爲零時，中央處理器的CARRY旗號會設爲1。 00＝無錯誤。 01＝未使用（語言未適用）。 02＝不正確的呼叫型。 03＝不正確的次呼叫（大於04） 04＝有寫入保護（WRITE PROTECTED） 05＝資料結束（END OF DATA） 06＝未找到檔案（若X＝0則爲它分配位置） 07＝不正確册號數 08＝磁碟輸出入錯誤

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11
OPEN	01		資料錄長或0000		V	D	S	檔案型式	檔案名稱之位址		返回碼（輸出）		檔案處理器工作區緩衝器之位址		T/S表緩衝器之位址		資料段落緩衝器之位址	
CLOSE	02																	
READ	03	次碼	資料錄號數		再偏移量		資料區域長度		第一冊之值	資料區域位址								
WRITE	04																	
DELETE	05				V	D	S		檔案名稱之位址									
CATALOG	06					D	S											
LOCK	07				V	D	S		檔案名稱之位址						T/S表位址			
UNLOCK	08				V	D	S											
RENAME	09		新名稱之位址		V	D	S											
POSITION	0A		資料錄號數		再偏移量													
INIT	0B	DOS頁號數			V	D	S											
VERIFY	0C				V	D	S		檔案名稱之位址						T/S表之位址		資料緩衝器之位址	

圖6-2 檔案處理器參數表所須之輸入

V 冊號數
D：推動器號數
S：段落號數



呼叫檔案處理器

09 ＝磁碟已滿。

0A＝檔案已鎖制。

0B　　未使用

0C/0D　45個拜緩衝器的位址。在呼叫檔案處理器之後，此緩衝器保存了檔案處理器的狀態（STATUS）。此區域稱爲檔案處理器工作區（WORKAREA），它不須使用者預先設定，使用者只要留下空間並訂定其位址（低拜/高拜）。

0E/0F　256拜緩衝器。檔案處理器用它來保存開檔案（OPEN FILE）的"T/S表"（軌道/段落表）。呼叫者不須預先設定此緩衝器。

10/11　256拜緩衝器。檔案處理器用它來保存資料段落。呼叫者不須預先設定它。

五、呼叫型式之下的檔案處理器參數表

開檔（OPEN）　定置或建立一個檔案。每個"開檔"之後應跟隨一"定位"（POSITION）。

輸入：拜00　　01

02/03　固定資料錄長度（RECORD LENGTH）（或0000 表可變資料錄長度）。

04　冊號數（或00表任意冊）

05　推動器號數（01或02）

06　槽路號數（01至07）

07　檔案型式（僅使用於新檔案）

$00 = TEXT（主文檔）

$01= INTEGER BASIC（整數BASIC檔）

$02= APPLESOFT BASIC（蘋果BASIC檔）

$04= BINARY（二進位檔）

$08= RELOCATABLE（可重定位檔）

$10= S TYPE FILE（S型檔）

$20= A TYPE FILE（A型檔）

$40= B TYPE FILE（B型檔）

08/09 檔案名稱之位址（30個文字的緩衝器）（低拜/高拜）。

0C/0D 檔案處理器的工作區緩衝器之位址。

0E/0F T/S表的段落緩衝器之位址。

10/11 資料段落緩衝器之位址。

輸出：拜 07 開檔案的檔案型式。

0A 返回碼。

閉檔（CLOSE）寫出最後的段落並更新目錄（CATALOG）。每個OPEN事件須以CLOSE結束之。

輸入：拜 00 02

0C/0D 檔案處理器的工作區緩衝器之位址。

0E/0F T/S表的段落緩衝器之位址。

10/11 資料段落緩衝器之位址

輸出：拜 0A 返回碼。

讀（READ） 讀入單一或一群拜到記憶體
寫（WRITE）寫出單一或一群拜到磁碟

輸入：拜00 03（讀），04（寫）。

01 次碼（SUBCODE）

00 —無作用。

01 —讀或寫一拜。

02 —讀或寫一群拜。

03 —定位後讀或寫一拜。

04 —定位後讀或寫一群拜。

02/03 （次碼爲03或04者）資料錄號數（RECORD NUMBER）。

04/05 （次碼爲03或04者）拜偏移量（BYTE OFF-SET）。

06/07 （次碼爲02或04者）讀或寫拜的數量。
註："寫"時，此值須比實際長度小1。

08/09 （次碼爲02或04者）被"寫"的拜群位址，或"讀"入拜群的緩衝器位址。

08 （"寫"並次碼爲01或03者）單一寫入拜的值

0C/0D 檔案處理器的工作區緩衝器之位址。

0E/0F T／S表的段落緩衝器之位址。

10/11 資料段落緩衝器之位址。

輸出：拜02/03 現在檔案所在位置的資料錄號數（RECORD NUMBER）

04/05 現在檔案位置的拜偏移量"。

08 （"讀"且次碼爲01或03時）所讀的單一拜的值。

0A　　返回碼

*現在檔案位置已更新為指向已讀或寫的資料的下一個拜位置

除檔（DELETE）定置及除去一檔案，空出所佔的段落。

輸入：拜00　　05

（其他參數與"開檔"的相同）

輸出：拜0A　　返回碼

目錄（CATALOG）列出目錄表於輸出裝置

輸入：拜00　　06

05　　推動器號數

06　　槽路號數

0C/0D　檔案處理器的工作區緩衝器之位址。

輸出：拜0A　　返回碼

鎖制（LOCK）鎖住一檔案

輸入：拜00　　07

（其他參數與"開檔"的相同）

輸出：拜0A　　返回碼

解制（UNLDOCK）將已鎖住的檔案解鎖

輸入：拜00　　08

（其他參數與"開檔"的相同）

輸出：拜0A　　返回碼

重定名(RENAME) 重定檔案名稱

輸入:拜00　　09

　　02/03　　新檔案名稱的位址(含30個拜,即30個字數)(其他參數與"開檔"的相同)

輸出:拜0A　　返回碼

定位(POSITION)　計算一記錄(RECORD)的位置,及/或檔案的拜偏移量以指出此檔案中下一個讀或寫的位置。在第一個"讀"或"寫"之前必須做定位工作(用此"定位"程式或由隱含於"讀"或"寫"內的次呼叫(SUBCODE)做之皆可)。往往定位於檔案起始位置,若是如此,輸入拜02到05都設爲零。

輸入:拜00　　0A

　　02/03　　相關的資料錄號數(RELATIVE RECORD NUMBER)(具固定長度資料錄的檔案,或存零的可變長度資料錄的檔案)。檔案的第一個資料錄爲資料錄0000。

　　04/05　　指入一資料錄或整個檔案(資料錄爲零表整個檔案只有一個資料錄之時)的相關拜偏移量。

　　0C/0D　　檔案處理器的工作區緩衝器之位址。

輸出:拜0A　　返回碼。

起始設定（INIT）起始設定一副磁碟。格式化並複製DOS 的第0至2軌道進入新磁碟，同時亦建入VTOC及目錄（CATALOG）。但HELLO程式並未存入。

輸入：拜	00	0B
	01	複製入新磁碟的DOS 映像第一頁。在48K系統中此值爲$9D
	04	新磁碟的册號數。
	05	推動器號數（01或02）
	06	槽路號數（01到07）
	0C/0D	檔案處理器的工作區緩衝器之位址。
輸出：拜	0A	返回碼。

查證（VERIFY）讀出每一段落以證實檔案中無不良段落。

輸入：拜	00	0C
	（其他參數與"開檔"的相同）	
輸出：拜	0A	返回碼。

六、DOS緩衝器

呼叫檔案處理器以保存記憶體時，須使用到一個DOS 緩衝器。這個DOS 緩衝器含有檔案處理器會利用到的三個緩衝器（檔案處理器工作區，T/S表的段落，及資料段落），30個拜的檔案名稱緩衝器，及許多鏈結指標。整個DOS 緩衝器共佔記憶體595個拜。第一個DOS 緩衝器的位址存在DOS 的第一拜中（48K系

統中爲$9D00)。下一個DOS 緩衝器的位址則存在第一個DOS 緩衝器中，如此繼續下去形成一緩衝器鏈（CHAIN）。其中最後一個緩衝器的鏈結（LINK）位址存零。若緩衝器未爲DOS 所用，其檔案名稱欄的第一拜內存$00，否則便爲開檔的名稱的第一字。組合語言程式師應依此規則運用，以免DOS 重覆使用緩衝器。

檔案名稱須先存入緩衝器以取得對緩衝器唯一使用權，並於使用完畢後將$00存入檔案名稱的第1拜供DOS 將來用之。若不如此，最後DOS 會耗盡可用的緩衝器，甚至拒絕執行（CATALOG）命令。FIG 6.3是MAXFILE 3的緩衝器結構圖。DOS 緩衝器格式如下：

DOS緩衝器格式

拜	說 明
000/0FF	資料段落緩衝器（長爲256個拜）。
100/1FF	T/S表段落緩衝器（長爲256個拜）。
200/22C	檔案處理器的工作區緩衝器（長爲45個拜）。
22D/24A	檔案名稱緩衝器（長爲30個拜）。

第一個拜顯示出此DOS 緩衝器是否正使用中，若第一個拜內容爲$00表此緩衝器空着未用。

24B/24C	檔案處理器的工作區緩衝器之位址（低拜/高拜）。
24D/24E	T/S表段落的緩衝器位址。
24F/250	資料段落緩衝器位址。
251/252	緩衝器鏈的鏈結（LINK），指向下一個緩衝器的檔案名稱欄之位址。若本緩衝器已是鏈中最末一個緩衝器，則此處塡入零。

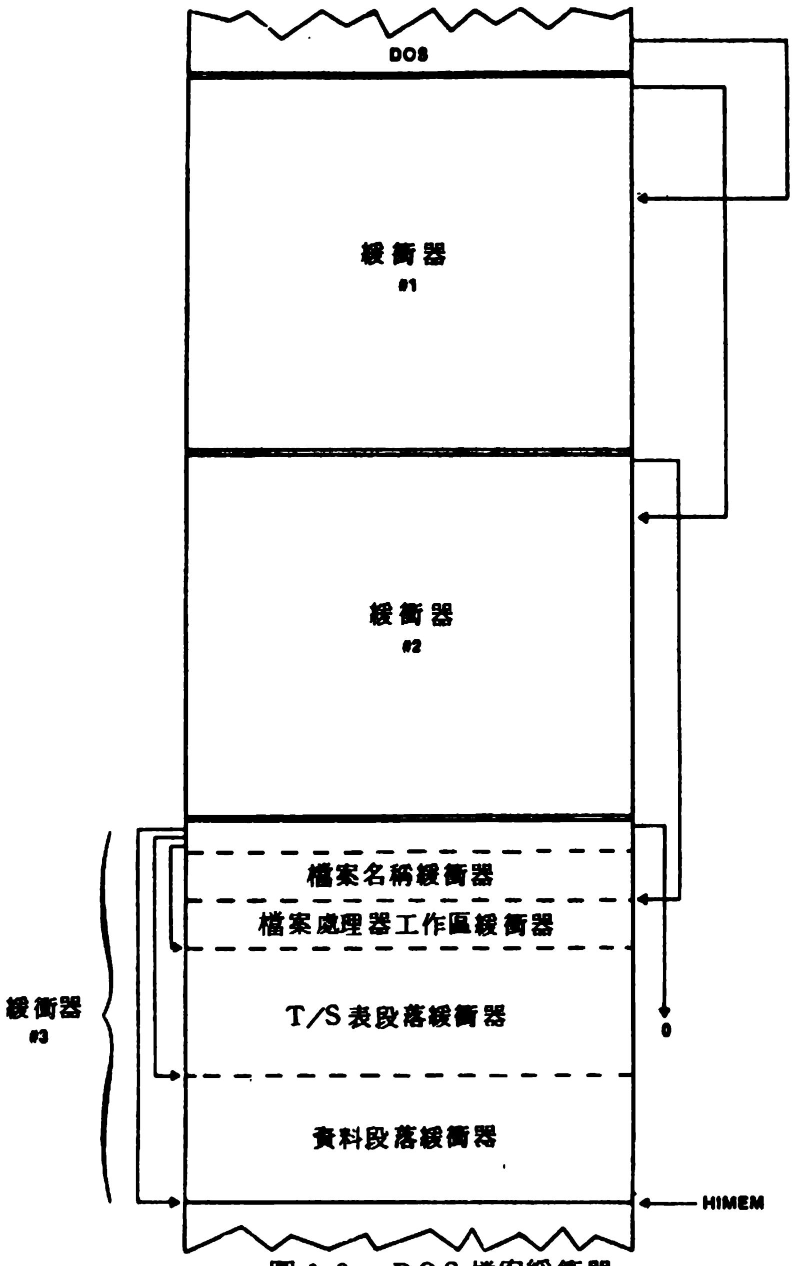


圖 6.3　DOS 檔案緩衝器

七、檔案處理器工作區

檔案處理器工作區含有許多變數，這些變數組成了檔案處理器作"開檔"時所須的所有資訊。每當檔案處理器作完一呼叫的工作後，便將它所有的重要變數複製入使用者給定的檔案處理器工作區緩衝器。到下回呼叫檔案處理器時，第一件事就是將檔案處理器工作區緩衝器的內容重新複製入檔案的內容重新複製入檔處理器工作區的變數中，以再度處理上次呼叫之後離開的檔案。平常程式設計師不須為此工作區的內容操心，因大多數有用的資訊都出現在參數表中。只是有時它可以使人對開檔案更加了解。以下為檔案處理器工作區的格式：

檔案處理器工作區之格式

拜	說　明
00/01	檔案的第一個T／S表所在的軌道／段落。
02/03	檔案的現在（CURRENT）T／S表所在的軌道／段落
04	旗號： 80＝T／S表之內容已更改，須重寫入磁碟中。 40＝資料緩衝器內容已更改，須重寫入磁碟中。 02＝册的自由空間映圖（VOLUME FREE-SPACE MAP）已更改，須重寫入磁碟。
05/06	現在（CURRENT）資料段落所在的軌道／段落。
07	目錄中檔案進入點的"段落偏移量"（SECTOR OFFSET）。
08	目錄段落中（07所指的段落）檔案進入點的"拜偏移量"（BYTE OFFSET）。
09/0A	一個T／S表所能表示的最大資料段落數。

0B/0C　現在的T/S表中第一個段落的偏移量。

0D/0E　現在的T/S表中最後一個段落的偏移量。

0F/10　前一次所讀的相關段落號數。

11/12　以"拜"爲單位的段落大小（256）

13/14　相關段落群中現在的位置（現在指向那個段落）。

15　此段落中現在的拜偏移量。

16　未使用

17/18　固定資料錄長度。

19/1A　現在資料錄號數。

1B/1C　指入現在資料錄的拜偏移量。

1D/1E　以"資料錄"爲單位的檔案的大小。

1F　本軌道的下一個配置（ALLOCATE）的段落。

20　正受配置的現在軌道。

23/24　本軌道上可利用的段落的比次對映（BIT MAP）。

25　檔案型式。00,01,02,04=T（TEXT），I（INTEGER），A（APPLESOFT），B(BINARY)。(80表已鎖制)

26　槽路號數乘16（如$60＝第6號槽路）。

27　推動器號數（01或02）

28　册號數（以補數型式表示），

29　軌道。

2A/2C　未使用。

八、共通法則

使用DOS時，以下數組程式可供利用。

定置一FREE DOS緩衝器：

以下副程式用來定置（LOCATE）一未配置的DOS 緩衝器供檔案處理器使用。

```
FBUFF   LDA   $3D2      定置DOS 載入指標（LOAD
        STA   $1         POINT）
        LDY   #0
        STY   $0
*
GBUF0   LDA   ($0),Y  定置下一個DOS 緩衝器。
        PHA
        INY
        LDA   ($0),Y
        STA   $1
        PLA
        STA   $0
        BNE   GBUF      取得一個緩衝器
        LDA   $1
        BEQ   NBUF      無FREE  BUFFER可用
*
GBUF    LDY   #0        取檔案名稱
        LDA   ($0),Y
        BEQ   GOTBUF  是未使用的緩衝器（FREE）、
        LDY   #36       已使用的緩衝器（NOT  FREE）
```

```
        BNE     GBUF0   去取下一個緩衝器。
*
GOTBUF  CLC             顯示已取得一個自由的（FREE）
                        緩衝器。
        RTS             返回呼叫者。
NBUF    SEC             顯示未取得自由的緩衝器。
        RTS             返回呼叫者。
```

判定正處於那一DOS 版本上：

有些程式含有與DOS 版本有關的數碼，因此DOS 版本須事先檢查過。

```
        CLC
        LDA     #0      加$16BE於DOS 載入指標
        ADC     #$BE    （LOAD POINT）
        STA     $0
        LDA     $302
        ADC     #$16
        STA     $1
        LDY     #0
        LDA     ($0),Y  取得DOS 版本的號數（2或3
```

判定DOS 是否在機器中：

以下指令須在呼叫RWTS 或檔案處理器之前用之，以確定DOS 已存入機器中。

```
        LDA    $3D0      取JMP指令碼。
        CMP    #$4C      檢查是否爲跳越指令。
        BNE    NODOS     不是，DOS仍未存入記憶體
```

判斷選用的BASIC種類：

許多程式的執行要靠INTEGER BASIC ROM或APPLE-SOFT ROM。以下程式可以檢查出正使用的BASIC種類，並可選擇所須的BASIC種類。首先載入一代表所須BASIC種類的數碼，INTEGER BASIC爲$20，APPLESOFT BASIC爲$4C，例如選用APPLESOFT時：

```
        LDA    #$4C      APPLESOFT的數碼
        JSR    SETBSC    呼叫副程式
        BNE    ERROR     語言尚未適用
         :
SETBSC  CMP    $E000     檢查是否爲正確的BASIC
        BEQ    RTS       是
        STA    $C080     非，選擇ROM卡
        CMP    $E000     再檢查看是否取得正確的BASIC
        BEQ    RTS       是
        STA    $C081     否，再試ROM卡
        CMP    $E000     再檢查，現在如何？
RTS     RTS              無條件返回
```

檢視是否正執行BASIC程式：

用下列程式檢視BASIC程式是否正在進行，或是處於立即命令模式(IMMIDIATE COMMAND MODE)。

使用INTEGER BASIC時——

```
LDA   $D9
BMI   EXEC     程式正執行中
BPL   NOEXEC   程式未執行
```

使用APPLESOFT BASIC時

```
LDX   $76      取得線號數(LINE NUMBER)
INX
BEQ   NOEXEC   程式未執行
LDX   $33      取得PROMPT
CPX   #$DD     PROMPT是"]"嗎?
BEQ   NOEXEC   是，則程式未執行
BNE   EXEC     否，則程式正執行中
```

第七章 依賴客要求修改DOS

儘管DOS 提供了極多的功能給予BASIC 及組合語言程式設計師使用，有時顧客仍會要求對DOS 做些變更。因DOS 經常有新版本推出且每數個月對DOS 做許多修改的工作亦是沉重的負擔，所以修改DOS 的要求只在絕對必要的情形下才會被接受。大量修改DOS 必須澈底了解每個變更的完全函意及所有的牽連，否則很可能使系統變得不可靠。

一、副磁碟及主磁碟的修補

一般修改DOS 的程序包含"修補（PATCHING）"DOS 的機器語言碼。當決定了所須要的修改後，DOS 的一些指令被另外一些所取代以修正程式。有三種修改DOS 的方式可用。

1. 只修補在記憶體中的DOS 。這種修改在REBOOT 之後便失效。

2. 執行第一種方式所修補的DOS 並以它起始設定一磁碟以得到DOS 永久的修改。就是建立（CREATE ）一磁碟，將DOS 複製在它第 0，1，2 軌道上。每當BOOT新建磁碟，修補過的DOS

版本便載入記憶體。往後在此磁碟上所建的任何副磁碟也都擁有修補過的DOS 版中。

3. 直接修補主磁碟。這種方式較麻煩。要利用ZAP 程式直接修補主磁碟前三軌道中的DOS 映像，或者利用MASTER CREATE 將已修改過的DOS 複製品寫入新磁碟。下列程序執行此工作：

BLOAD MASTER CREATE

進入監督程式（CALL－151 ）

將＄4C 存入＄80D 位置中（ 80D：4C ）

執行MASTER CREATE（800G ）

當MASTER CREATE 將DOS 映像載入記憶體後便跳出。然後使用者可以利用監督程式修改DOS 映像。MASTER CREATE 首先將DOS 的BOOT2 （ RWTS）首先自記憶體＄1200 載入，接著將DOS 主體自＄1C00載入。

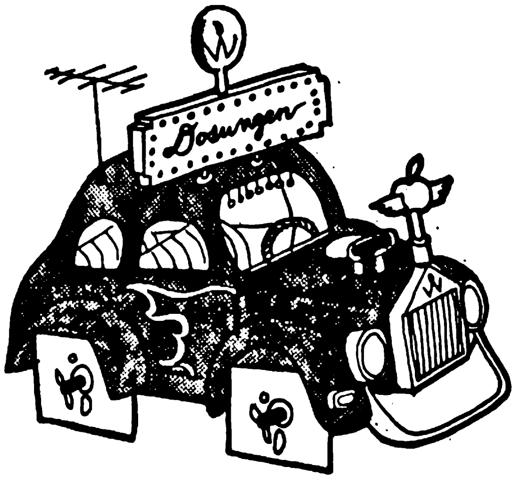
修補完後重入MASTER CREATE ＄82D位置（ 82DG ）

MASTER CREATE 正確地執行完後，磁碟就完成了修補的工作。

這程序可用在DOS 3.2，3.2.1 及 3.3版本上。

二、語言卡的避免重載入

增訂在DOS3.3之中有個較困擾的地方。BOOT2 機器碼中有個修補，是將語言卡第一個拜存入二進位制"零"，以使得不論是否爲開機狀態，每次BOOT 都要重載入BASIC（ INTEGER或APPLESOFT ）。開機時這個修補沒有作用，因爲語言卡上第一



顧客化磁碟

拜並未呈現於DOS 之前供判定爲BASIC ，不論如何此時BASIC都須重載入記憶體。但在其後的BOOT中，往往記憶體已有完整的BASIC ，而此修補在BOOT磁碟時仍造成LANGUAGE NOT AVAILABLE（語言仍未適用）的錯誤信號。DOS3.3的這個修補原是考慮到可能發生類如PASCAL 語言的第一拜恰巧同於BASIC語言的第一拜之情形，這結果會在DOS 中造成怪異的現象。若使用者在交換使用DOS 及其他系統時總是關機再開機，可將這個修補以下列方法拿掉不用：

將$BFD3（48K）三拜的STA 指令改成三個NO OP 指令：

BFD3：EA EA EA

以此修改過的DOS 起始設定一副磁碟可得新修補DOS 磁碟。此STA 指令在32K DOS系統中位於7FD3，在16K DOS系統中

位於 3FD3 。

三、將程式插入DOS及DOS緩衝器之間

有時須要一個安全的區域存常用的機器碼程式(如:即表機推動器 (PRINTER DRIVER) 等) 使其不受BASIC及DOS 干擾亦不受DOS 冷啓動 (COLDSTART) 影響。短於 200 拜的程式可將它存入自 $300 開始的位置。若程式大於 200拜，最好將它"塞"入DOS 及DOS 緩衝器之間 (此程式須爲可重定位 (RELO -CATABLE) ，並可在新位置執行) 。首先將程式載入到記憶體低位置區再將它複製到低於 $ 9D00 (48K) 及緩衝器上方，然後更改 $9D00 內容，使第一個緩衝器的位址指向插入程式的下方 (須多空下存檔案名稱及鏈結欄 (LINK FIELD) 所佔的38拜位置) ，並將程式跳至$ 3D3 (DOS 冷啓動)使DOS在插入程式的下方另建緩衝器。在下一次 BOOT之前，DOS會把插入程式所佔空間忽略掉。因HIMEN 低於DOS 緩衝器，BASIC 不會使用那裡的記憶體。

四、BRUN或EXEC命令執行招呼(HELLO)檔

當DOS 完成了BOOT並進入記憶體後，它執行了在檔案名稱緩衝器中HELLO檔案的RUN 命令 (其左爲將DOS 寫入磁碟的 INIT命令) 。以下述方法修補DOS 可將RUN 命令改爲BRUN 或EXEC 。

9E42：34（改爲BRUN）

或9E42：14（改爲EXEC）

五、列出長目錄時去除“暫停”的功能

當下一CATALOG 命令於具衆多檔案的磁碟時，每打滿一次檔案名稱於整個顯示幕，DOS 會自動暫停動作以便使用者充分時間看清楚。其後任意按下一鍵可使CATALOG 繼續動作。若欲除掉此"暫停"功能，可作下列修補：

AE34：60

第八章 DOS的程式邏輯

在這一章裏，將詳細說明DOS 程式本身的動作，幫助讀者做深入的瞭解，以便對其所提的功能做更佳的運用。對每一個副程式及變數或常數群均分別按其儲存位址加以說明。較積極的讀者可能希望將DOS 解碼成組合語言予以印出，而直接將這裏的說明加註上去。此處說明中的位址乃是配合DOS 3.3 及有48K 主磁碟的DOS。使用槽路設爲6 。除非特別用 $ 符號標明，長度皆以十進位，而位址皆以十六進位表示。

APPLE DISK II 控制卡的ROM－BOOTO

位　址

C600-C65B　此常式在新啓動一磁碟時，須首先執行。經由命令PR#6或C600G或6 Control-P 而取得控制權。

動態地在$356-$3FF 之中建立一個轉換表，以便把磁碟碼轉換成六比次碼。

叫用一個僅讀記憶體監督程式中的RTS 指令，從堆疊（STACK）中取出返回位址，來找出僅讀記憶體控制器的位址。

利用此位址對$Csxx 的偏移以決定此推動器的槽路號數。

保存槽路號數乘以16的值（$s0）

清除磁碟輸入/輸出門，設定讀取模式，選擇1號推動器，將磁碟推動器開啓。

將探臂拉回定位，重新調整到軌道0。

設好參數以便將軌道0中的段落0讀入$800中。

進入段落讀取副常式（C65C）中執行。

C65C-C6FA　此副常式將$41所存的軌道中$3D所指的段落讀入$26,$27 所指的位址中。

尋找磁碟上的段落位址標頭D5/AA/96。

若找到D5/AA/AD，且需要該段落中的資料，則進入C6A6。

C683　處理一個段落位址塊（Block）。

從磁碟上讀取三個雙拜，經組合後得出册、軌道及段落號數，這個段落就是即將從磁碟上讀取的。

將軌道號碼存在$40。

比較找到的段落及軌道號數與所要的段落及軌道號數。
若不相符，則跳回C65C。
否則，若段落正確，則跳到(C65D)找出該段落資料本身。

C6A6 處理段落資料塊。
將二次資料85個拜讀入$300-$355中。
將一次資料256個拜讀入$26，$27所指的位址中
驗證資料的偵錯數（Checksum）是否正確。
若不對，則跳到C65C。
將一次及二次資料一併經拆組轉換後放入一次資料緩衝器（$26，$27）。
將$27（存資料的頁位址）及$3D（欲讀取的段落）中的值遞加，並與$800 中的值比較，檢查是否尚有段落要讀。
若還有，則將SLOT＊16值載回，並跳回C65C 讀取下一個段落（此動作不用於載入DOS，但可用於從一個BASIC磁碟載入）。
否則，進入$801 開始執行第二階段的啓動工作（Bootstrap）。

第一個RAM靴帶起動載入程式－Boot 1（起動程式1）

位　址

0801-084C 此程式將第二隨機讀寫式載入程式（RAM LOADER）（即Boot2），包括RWTS，載入記憶體中並跳去執行。

若這不是第一次進入Boot1 ，則進入$81 F。
取得SLOT*16值並右移得到槽路號數。
建立螢讀記憶體段落讀取副常式的位址（此處爲C65C），並存入$3E，$3F中。
取出第一頁記憶，其中$8FE存有Boot2的開始位置，加上$8FF存放的Boot2段落長度值，將此值設爲讀取的第一個位址（最先讀取最後一頁）。

081F　取得欲讀入的段落，若爲0，則進入$839。
將理論上的段落號數轉換爲實際的段落號數（參照$84D的偏距表）。
將段落的理論號數遞減（$8FF），以供以下反覆執行。
設好ROM 的副常式（C65C）的參數，並跳去執行，當段落被讀入時，再回到$801。

0839　調整在8FE 的頁數，以找出Boot2的進入點。
叫用監督程式執行PR#0 和1N#0 。
重新設定監督程式（TEXT模式，標準顯示窗等）再取得SL0T*16 值並進入Boot2（在主磁碟爲$3700，在重定位的最後位置爲$B700）。

DOS 3.3主要常式

位　址

9D00-9D0F　可重定位的位址常數。

9D00　第一個DOS 緩衝器中名稱欄的位址。

9D02	DOS 鍵盤截斷處理常式的位址。
9D04	視訊截斷處理常式位址。
9D06	主要檔名緩衝器的位址。
9D08	第二檔名（RENAME）緩衝器的位址。
9D0A	LOAD 命令中範圍長度參數的位址。
9D0C	DOS 載入位址（$9D00）的位址。
9D0F	檔案處理程式參數表的位址。
9D10-9D1C	DOS 視訊（CSWL）截斷狀態處理常式位址表。狀態是用來推動DOS 處理經由BASIC的PRINT敍述句送出的DOS 命令，而這個表包含了處理這些狀態（0～6）的常式位址。
9D1E-9D55	命令處理常式的進入表。這個表含了每一個DOS命令處理副程式的位址，標準順序如下：

```
INIT      A54F
LOAD      A413
SAVE      A397
RUN       A4D1
CHAIN     A4F0
DELETE    A263
LOCK      A271
UNLOCK    A275
CLOSE     A2EA
READ      A51B
EXEC      A5C6
WRITE     A510
POSITION  A5DD
OPEN      A2A3
APPEND    A298
RENAME    A281
CATALOG   A56E
MON       A233
NOMON     A23D
PR#       A229
IN#       A22E
MAXFILES  A251
FP        A57A
INT       A59E
BSAVE     A331
BLOAD     A35D
BRUN      A38E
VERIFY    A27D
```

9D56-9D61 現用的BASIC 進入點向量表。這些位址由DOS安排，供目前使用中BASIC 執行之用。

9D56 BASIC 的CHAIN 進入點位址。

9D58 RUN 的位址。

9D5A 錯誤處理常式的位址。

9D5C BASIC 冷起動常式的位址。

9D5E BASIC 暖起動常式的位址。

9D60 BASIC 重定位位址（僅供APPLESOFT 之用）。

9D62-9D6B INTEGER BASIC 進入點向量的映像（IMAGE）。
若INTEGER BASIC 要成為現用的時，則將此映像抄錄到 9D56 。

9D6C-9D77 ROM版APPLESOFT 進入點向量的映像。

9D78-9D83 RAM版APPLESOFT 進入點向量的映像。

9D84-9DBE DOS冷起動進入常式。
取得槽路和推動器號數並保存起來做為命令鍵語的自然值（Default）。
將APPLESOFT ROM或INTEGER BASIC 的進入點向量抄錄到目前的BASIC 進入點向量中。
記住何者BASIC 是目前現用的。
進入 9DD1。

9DBF-9DE9 DOS 暖起動進入常式。
取得記住的BASIC 型式，且必要時設定僅讀記憶體卡片（A5B2）。

9DD1 記住進入時是冷起動或暖起動。
叫用A851 更換DOS 鍵盤和視訊截斷向量。

設定NOMON C,I,O。
設定視訊截斷處理常式的狀態爲0。
冷或暖起動現用的BASIC(跳出DOS)。
(DOS在下次當BASIC印出輸入催促符號(PROMPT)時再度取得控制。)

9DEA-9E50 DOS第一次進入的處理。在DOS冷起動之後,BASIC第一次要求鍵盤輸入資料時,此常式即被鍵盤截斷處理常式叫用。
若現用的BASIC爲RAM APPLESOFT,則將其進入點向量抄錄到現用BASIC進入點向量,並將第一檔名緩衝器清爲空白,以致不會執行HELLO檔。
設定MAXFILES的自然值爲3。
叫用A7D4建立DOS檔案緩衝器。
若有EXEC正使用中,則關閉EXEC檔。
設定視訊截斷狀態爲0,並叫用A75B以標示暖起動狀態。
若上一個執行的命令不是INIT(即此DOS並非剛起動的),則進入9E45。
否則,抄錄DOS跳越向量映像到$3D0~$3FF。
將$3F2,$3F3指向DOS暖起動常式。
在RESET命令處理常式的位址改變之後,設定AUTOSTART ROM的POWER-UP拜。
設定RUN的命令索引(以執行HELLO檔),並進入A180執行。

9E45 檢查是否有等待中的命令。

若有，則進入A180 去執行。否則，跳回主叫方。

9E51-9E7F 此為DOS 第三頁跳越向量的對映，由前一常式將其抄錄到 $3D0-$3FF。對其內容，第五章有較詳細的說明。

9E81-9EB9 DOS 鍵盤截斷處理常式。
在進入DOS 時，叫用 9ED1 以保存暫存器的值。
若不是正在冷起動或讀取磁碟檔時，則進入 9E9E。
取得A暫存器在進入時的值並回應於顯示幕上（將閃動的游標抹掉）。
若是在讀取狀態（正在讀取檔案），則進入 A 626，從磁碟檔讀取下一個拜。
否則，叫用 9DEA處理第一次的進入。
將顯示幕上的游標移向下一位置。

9E9E 若正在執行EXEC，則叫用A682 ，從EXEC 檔讀取下一個拜。
設定視訊截斷狀態為 3（輸入回應）。
叫用9FBA存回暫存器在進入DOS 時保存起來的值。
叫用實際鍵盤輸入常式。
將輸入字母保存起來，使得可以將其存回A暫存器中。
對新的X暫存器值也同樣處理。
經由9FB3 跳出DOS。

9EBA-9EBC 跳到實際KSWL 的處理常式。

9EBD-9ED0 DOS 視訊截斷處理常式。
在進入DOS 時叫用 9ED1 保存所有暫存器的值

取得視訊截斷狀態，並當作尋找截斷處理常式表（9D10 ）的索引，找到之後，即進入該處理程式去執行，同時傳送要印出的字母。

9ED1-9EEA 截斷時保存暫存器的共用常式。
保存A，X，Y和S暫存器的值於AA59-AA5C
若是在DOS 中，則將實際I／0處理常式（KSWL和CSWL）存回$36-$39之中。
跳回主叫方。

9EEB-9F11 狀態0輸出處理常式——一行開始——
若RUN 命令被截斷（裝入RAM APPLESOFT ），則進入9F78以繼續完成。
若"讀取"旗號爲ON（正在讀取檔案）且輸出是一個"？"符號（BASIC 輸入指示），則進入狀態6將其略過。
若"讀取"旗號爲ON ，並輸出爲催促符號（$33 ），則進入狀態2，將此行略過。
設定狀態爲2（略過非DOS 的命令）以備萬一。
若輸出字母不是 Control-D，則跳到狀態2。
否則，設定狀態爲1（收集可能的DOS 命令），設定行索引爲0，並進入狀態1。

9F12-9F22 狀態1輸出處理常式——收集DOS 命令——
利用行索引，將字母存入$200 的輸入緩衝器。遞增行索引。
若該字母不是"回車"（Carriage Return ），則經由9F95 跳出DOS （若MON I，則回應於顯示幕上）。

否則，進入$9FCD 的命令掃瞄常式。

9F23-9F2E 狀態2輸出處理常式——跳過非DOS 命令——

若字母不是"回車"，則經由9FA4(將字母回應於顯示幕上)跳出DOS 。

否則，將狀態設爲0並經由9FA4跳出DOS 。

9F2F-9F51 狀態3輸出處理常式——INPUT敍述句處理常式

若INPUT結束時，將狀態設爲0。

若字母不是"回車"，只要EXEC實際上沒有NOM-ON I條件，則將其回應於顯示幕上。之後，不管字母爲何，皆跳出DOS (KSWL會設定狀態爲3。)

否則叫用A65E檢查BASIC是否正在執行程式或在直接模式(Immediate)。若EXEC正在執行或BASIC正在直接模式，則進入狀態1去收集可能的DOS 命令。

否則，跳出DOS ，在需要時回應字母。

9F52-9F60 狀態4輸出處理程式——WRITE資料到一檔——

若字母爲"回車"，則設定狀態爲5(開始寫一行資料)。叫用A60E將拜寫入磁碟檔。

跳出DOS並當MON 0時回應於顯示幕上。

9F61-9F70 狀態5輸出處理常式——開始WRITE一行資料—

若字母爲Control -D，則進入狀態0，立即跳出寫入模式。若字母爲LF(進一行)，則將其寫出並跳出，停留在狀態5。

否則，設定爲狀態4，並進入狀態4。

9F71-9F77 狀態6輸出處理常式——跳過催促信號——

設定狀態爲 0。

經由 9F9D 跳出DOS （若MON I時則回應）。

9F78-9F82 完成被APPLE RAM LOAD 動作（載入APPLE RAM ）所中斷的RUN 命令。

清除" RUN interrupted"（RUN 被中斷）旗號

叫用A851 更換DOS CSWL/KSWL截斷向量。

進入A4DC完成RUN 命令。

9F83-9F94 DOS 命令掃瞄常式的跳出BASIC常式。

若命令的第一字是Control-D，則進入回應跳出常式（ 9F95 ）。

否則，設定好傳送一行長度爲 0 的命令（僅有"回車"）使BASIC不會收到DOS 命令。往下執行回應跳出常式。

9F95-9FB0 根據條件將字母回應在顯示幕上，並跳出DOS 。

9F95 只在MON C設定時才回應，否則進入 9FB3 。

9F99 只在MON O設定時才回應，否則進入 9FB3 。

9F9D 只在MON I設定時才回應，否則進入 9FB3 。

9FA4 必然回應。

在進入DOS 時，叫用 9FBA存回暫存器的值。

叫用 9FC5 將字母回應於顯示幕上。

回應之後，保存暫存器的內容。

往下執行DOS 跳出常式。

9FB3-9FC4 DOS 跳出處理常式及暫存器的存回。

叫用A851 將DOS KSWL/CSWL截斷處理向量換回。

將S（堆疊）暫存器在進入DOS時所保存的值存回。

9FBA DOS 暫存器存回副程式。
存回暫存器在第一次進入DOS 時所保存的值，並跳回主叫方。

9FC5-9FC7 跳越到實際CSWL常式。

9FC8-9FCC 在顯示幕上跳過一行。
將"回車"（CR）符號存入A暫存器並叫用 9FC5 將其印出。

9FCD-A179 DOS 命令剖析常式。
設定命令索引爲-1（無命令）。
清除命令等待旗號（沒等待）。

9FD6 命令索引加1。
若第一字是Confrol-D，則跳過。
跳過空白，找到非空白的字爲止（叫用A1A4）。
利用目前的命令索引，將命令與位於A884 的命令名稱表中的名稱比較。
若不符合且尚有進入點可供比對，則跳回 9FD6。
若符合，則進入A01B。
否則，若命令在表中找不到，則檢查第一個字是否爲Control-D。
若是，則進入A6C4 印出"SYNTAX ERROR（句法錯誤）。
否則，叫用 A75B 來清除狀態及暖起動旗號並進入 9F95 回應命令後跳出。（即此一命令必屬BASIC，而非DOS 者）。

A01B 計算送進來的命令在運算元表中的索引。
叫用A65E 檢查是否有一個BASIC程式正在執行

若無，且命令不屬直接式（DIRECT TYPE）（根據運算元表可知），則進入A6D2 印出" NOT DIRECT COMMAND "（不是直接命令）。

否則，若命令是RUN ，則使催促符號（$33）不被印出。

核對運算元表，看命令中第第一個檔名是否爲有效的運算元。

若屬無效，則進入A0A0 。

否則，叫用A095 來清除檔名緩衝器。

叫用A1A4 來掃瞄命令，以找到下一個非空白的字，並將檔名運算元抄錄到第一個檔名緩衝器。

若一直是空白，則跳過一個逗點。

若第二個檔名合乎此命令要求，則再使用前面步驟將其抄錄到第二個檔名緩衝器中。

檢查這兩個檔名是否爲空白。

若命令中需要檔名而未予提供，則指出句法錯誤（SYNTAX ERROR ）或將指示傳送給BASIC （譬如LOAD時沒有運算元）。

如果兩個檔名都對，則進入A0D1 繼續。

A095 "將兩個檔名緩衝器清爲空白"副常式。

A0A0 指示尚未剖析過檔案名稱。

檢查運算元表是否需要位置性的運算元。

若無，則進入A0D1 繼續。

否則，叫用A189 來轉換數值運算元。

若該運算元被省掉，則指出"句法錯誤"（SYNTAX ERROR ）。

若轉換後的數值超過16，則指出"範圍錯誤"(RANGE ERROR)。

若應該是槽路號數而值超過7，則指出"範圍錯誤"。

若不是槽路號數而其值是0，則指出"範圍錯誤"(MAXFILES 0 乃非數值性)。

A0D1 設定鍵語(Keyword)運算元的自然值(V＝0，L＝0，B＝0)。

A0E8 求取行偏移量索引並跳過空白及逗點找到下一個非空白的字。

若該行尚未結束，則進入A01C。

檢查是否有該命令不允許的鍵語。

若無，則進入A17A 去處理命令。

A10C 在有效鍵語表中找出命令中的鍵語(A941)。

若不在表中，則指出"句法錯誤"(SYNTAX ERROR)。

在所找到的鍵語的旗號中，取得其比次位置。

若鍵語沒有運算元，則進入A164。

否則，在旗號中標示鍵語已找到。

將該鍵語中的數值性運算元加以轉換。

若屬無效，則指出"句法錯誤"。

檢查此數值是否在鍵語有效範圍表(A955)所指定的可接受範圍內。

將鍵語的值保留在從AA66 開始的鍵語數值表內。

繼續剖析下一個鍵語，進入A0E8。

A164 指示C，I或O鍵語已被剖析過。

適當地更正鍵語數值表中的MON 值。

繼續剖析下一個鍵語。進入A0E8。

A17A-A17F 叫出A180 處理命令，然後跳往9F83 送出回應後跳出。

A180-A192 執行命令。

重新設定視訊截斷處理狀態為0。

清除檔案處理程式的參數表。

利用命令索引，從9D1E的命令處理常式表中取得命令處理常式的位址，並進入執行。

命令處理常式會跳回原叫用程式

A193-A1A3 從一行命令中取出下一字，檢查是否為CR或逗點。

A1A4-A1AD 掃視命令行，跳過空白，找到非空白的字。

A1AE-A1B8 將檔案處理程式的參數表（B5BB）全清為0。

A1B9-A1D5 將命令中的數值運算元轉換為數值。根據是否有$符號而決定叫用A1D6（十進位）或A203（十六進位）。

A1D6-A202 十進位轉換副程式。

A203-A228 十六進位轉換副程式。

A229-A22D PR#n命令處理常式。

將已剖析好的數值載入，並經由僅讀記憶體監督程式的FE95 跳出。

A22E-A232 IN#n命令處理常式。

將已剖析好的數值載入，並經由監督程式中的FE8B跳出。

A233-A23C MON 命令處理常式。

將新的MON 旗號加在AA5E 的舊旗號上，然後跳

出。

A23D-A250　NOMON 命令處現常式。
若有C，則送出一個"回車"（CR）信號，因為該行雖經回應，但未送出"回車"。
將AA5E中有關的比次OFF 掉後跳出。

A251-A262　MAXFILES命令處理常式。
將任何現用的EXEC檔取消。
關閉所有已開啓的檔（A316）。
設定新MAXFILES值於AA57。
進入A7D4，重建DOS 檔案緩衝器之後跳出。

A263-A270　DELETE 命令處理常式。
載入檔案處理程式的刪除運算碼（05）。
叫用檔案處理程式的開檔推動常式（A2AA），執行刪除。
叫用A764 找出其執行刪除時所使用的檔案緩衝器並釋放之。
跳回主叫方。

A271-A274　LOCK命令處理常式。
載入檔案處理程式的鎖制運算碼（07）後進入A277。

A275-A27C　UNLOCK 命令處理常式。
載入檔案處理程式的解制運算碼（08）。

A277　叫出開檔推動常式（A2AA）以執行所要的功能。
經由關檔（A2EA）後跳回主叫方。

A27D-A280　VERIFY 命令處理常式。
載入檔案處理程式的查證運算碼（OC），並進入A277執行。

A281-A297 RENAME 命令處理常式。

將第二個檔案名稱的位址存入檔案處理程式的參數表中。

載入檔案處理程式的更名運算碼（ 09 ）。

叫用A2C8 的推動程式。

經由關檔（ A2EA ）後跳出。

A298-A2A2 APPEND 命令處理常式。

叫用A2A3 來開啓（ OPEN ）檔案。

逐拜地讀取檔案，到找到零爲止。

若附加（ APPEND ）旗號有設定，則在資料錄號數加1 後將旗號OFF掉。

叫用POSITION命令（ A5DD ）以跳出。

A2A3-A2A7 OPEN命令處理常式。

設定檔案型式爲" TEXT "。

進入A3D5開啓檔案。

A2A8-A2E9 命令處理常式中的檔案管理共用常式。

設定運算碼爲" OPEN "。

A2AA 若在命令中未給L值，則用0001 ，並將資料錄長度值存入參數表中。

A2C8 若檔案已開啓則關閉之。

是否尙有可用的檔案緩衝器？

若無，則指出"沒有可用的檔案緩衝器"（ NO FILE BUFFER AVAILABLE ）。

將 $40,$41 指向空閒的檔案緩衝器。

叫用A743將檔案抄錄到檔案緩衝器（配置之）。

將緩衝器指標抄錄到檔案處理程式的參數表（A741

中。
叫用A71A完成檔案處理程式中參數的填入。
設定運算碼於參數表中。
經由推動常式跳出。

A2EA-A2FB CLOSE命令處理常式。
若命令中不含檔名，則進入A316關閉所有的檔。
否則，從已開檔案的緩衝器中取得檔名（A764）
若無此檔，則跳回主叫方。
否取，關閉該檔，並釋放所用到的緩衝器(A2FC)
再回到CLOSE命令處理常式，確信該檔所開啓的
緩衝器都被釋放。

A2FC-A315 關閉檔案並釋放所使用的緩衝器。
若此緩衝器屬於EXEC(A7AF)，則將EXEC旗號
OFF掉。
將$00存入檔名欄，即可將該緩衝器釋放。
將檔案緩衝器的指標抄錄到檔案處理程式的參數表
中。
設定檔案處理程式的運算碼爲CLOSE。
從其推動常式跳出。

A316-A330 關閉所有已開的檔。
指到第一個檔案緩衝器（A792）。
進入A320。

A31B 指向鏈結的下一個緩衝器位址（A79A）。
若已到鏈結的結束，則跳回主叫方。

A320 此檔案緩衝器是否屬於EXEC？
若是，則略過而進入A31B。

該緩衝器是否在使用中（已開啓）？
若未開啓，則略過並跳到A31B。
否則，關閉此檔並釋放之（A2FC）。
進入A316 再做。

A331-A35C　BSAVE 命令處理常式。
確信命令中有A和L鍵語。
若沒有，則指出"句法錯誤"。
叫用A3D5，開啓並驗證此爲B型檔。
把鍵語A值寫入檔案的前兩個拜中。
把鍵語L值寫入檔案的次兩個拜中。
利用A值，將一段範圍的拜內容抄錄到檔案後跳出。

A35D-A38D　BLOAD 命令處理常式。
不管檔案型式爲何，均開啓之。
檢查是否爲B型檔。
若不是，則指出"檔案型式不符"（FILE TYPE MISMATCH）。
否則，開啓B型檔並測試其檔案型式（A3D5）。
從該檔的前兩個拜中讀取A值。
若命令中未提供A鍵語，則利用此處讀取的值。
讀取次兩個拜中的L值。
進入A471 從檔中讀取一段範圍的拜內容到記憶體中。

A38E-A396　BRUN 命令處理常式。
叫用BLOAD 命令處理常式，將檔案載入記憶體中。

更換DOS 的截斷處理向量。

跳掉A的位址開始執行二進位碼程式，以跳出DOS。

A397-A3D4 SAVE命令處理常式。

叫用AAB6以便知道現用的BASIC型式。

若爲INTEGER，則進入A3BC。

若爲APPLESOFT，則測試$D6 旗號，以檢查程式是否受保護。

若有保護，則指出"程式太大"(PROGRAM TOO LARGE)。

否則，開啓並測式是否爲A型檔(A3D5)。

計算程式長度(PGMEND-LOMEN)。

將此佔兩拜長的長度值寫入檔案中。

將LOMEN開始的程式映像當作一段範圍的拜內容，予以寫入後跳出(A3FF)。

A3BC 開啓並測試是否爲I型檔(A3D5)。

計算程式長度(HIMEN-PGMSTART)。

將此佔兩拜長的長度值寫入檔中。

將PGMSTART開始的程式映像當作一段範圍的拜內容，予以寫入後跳出(A3FF)。

A3D5-A3DF 開啓並測試檔案的型式。

設定所要的檔案型式於檔案處理程式的參數表中。

叫用A2A8 開啓檔案。

進入A7C4 檢查檔案型式。

A3E0-A3FE 寫入一個兩拜長的值到開啓的檔中。

將要寫入的值存入檔案處理程式的參數表中。

設定"寫入一拜"的運算碼。

叫用檔案處理程式的推動常式。
再叫用一次以寫入第二個拜，並跳回主叫方。

A3FF-A40F 讀／寫一段範圍的拜。
把範圍值的位址設定在檔案處理程式的參數表中。
設定讀／寫一段範圍的拜的運算次碼。
叫用檔案處理程式的推動常式。
關閉該檔。
由VERIFY 命令確信所寫入的資料正確後跳出。

A410-A412 指出"檔案型式不符"（FILE TYPE MIS-MATCH）。

A413-A479 LOAD命令處理常式。
關閉所有的檔（A316）。
開啓所要的檔。
是否爲A型或I型檔？
若不是，則指出"檔案型式不符"。
現用的BASIC是那一個？
若是INTEGER，則進入A450。
選定爲APPLESOFT BASIC(A4B1)。若RAM 版必須要執行。則此一叫用可能使得DOS失去控制權。
讀取檔案的前兩個拜，此兩個拜指出程式的長度。
將此長度加到LOMEN（程式開頭）以算出程式結尾。
程式結尾超過HIMEN嗎？
若超過，則關閉該檔並指出"程式太大"。
在變數指標中設定程式的結束和開始值。

	讀取從頭開始的一段程式。
	更換DOS 截斷處理向量（A851）。
	進入BASIC 的重定位常式，將RAM APPLE-SOFT程式轉換爲ROM ，需要時亦可從ROM 轉換爲RAM 。
A450	選定INTEGER BASIC（A4B1）。
	讀入程式的長度值（檔中最前面兩個拜）。
	計算程式開始位置（HIMEN-LENGTH）。
	若爲 0 或小於 LOMEN，則指出"程式太大"，並關閉該檔。
	設定程式的開始指標。
	將程式讀入記憶體的一段範圍內。
	跳回主叫方。
A47A-A4AA	從檔中讀取兩個拜（位址或長度）。
	設好"讀取兩個拜"的參數於範圍長度欄（AA60）
	叫用檔案處理程式的推動常式。
	若讀入的值恰爲長度，則將其存入檔案處理程式參數表中當作範圍長度。
A4AB-A4B0	關閉檔案值並指出"程式太大"。
A4B1-A4D0	選定所要的BASIC 。
	若所要的BASIC 目前已是現用的，則跳回主叫方。
	若是必須要使用APPLESOFT ，則保留目前的命令索引。
	若要INTEGER，則進入A59E 以選定之。
	否則，若要RAM APPLESOFT時，先將主檔案名稱抄錄到次緩衝器中保存。

進入A57A，選定APPLESOFT 。

A4D1-A4E4　RUN 命令處理常式。

若現用的BASIC是APPLESOFT ，則設定"RUN 被截斷"旗號，在APPLESOFT 載入後，RUN 可繼續完成。

叫用LOAD命令處理常式來載入程式。

在顯示幕上跳過一行。

換回DOS 截斷處理向量。

進入目前BASIC中的RUN 進入點。

A4E5-A4EF　INTEGER BASIC RUN 進入點截斷處理常式

刪掉所有的變數(相當於CLR)。

進入INTEGER BASIC中CHAIN 的進入點。

A4F0-A4FB　CHAIN命令處理常式。

叫用LOAD命令處理常式來載入程式。

跳過一行。

更換DOS 截斷處理向量。

進入目前BASIC 的CHAIN 的進入點。

A4FC-A505　ROM APPLESOFT的RUN進入點截斷處理。

叫用APPLESOFT 來清除所有的變數。

重新設定ONERR 。

跳到RUN 進入點。

A506-A50D　APPLESOFT RAM RUN 進入點截斷處理。

叫用APPLESOFT 來清除所有的變數。

重新設定ONERR 。

跳到RUN 進入點。

A510-A51A　WRITE 命令處理常式。

叫用READ/WRITE 共用常式（A526）。
設定CSWL狀態爲5（寫出模式的一行開始）。
跳出DOS （9F83）。

A51B-A525 READ命令處理常式。
叫用READ/WRITE 共同常式（A526）。
在狀況旗號中設定讀取模式旗號（AA51）。
跳出DOS （9F83）。

A526-A54E READ/WRITE 共同常式。
找出此檔的開檔緩衝器（A764）。
若未開啓，則將其開啓。
將檔案緩衝器位址抄錄到檔案處理程式的參數表中。
若命令中有R或B，則將其抄錄到參數表中，並向檔案管理程式發出POSITION的叫用。
跳回主叫方。

A54F-A56D INIT命令處理常式。
若命令中有V，則使用之；否則，使用254。
將DOS 的第一頁號數存入檔案處理程式的參數表中。
叫用檔案處理程式的推動常式來重新設定磁碟。
叫用SAVE命令，將問候程式（Greeting program）存入磁碟後跳出。

A56E-A579 CATALOG 命令處理常式。
使用CATALOG 的運算碼叫用檔案處理程式。
設定新的V值爲自然值，供以後的命令使用。
跳回主叫方。

A57A-A59D FP命令處理常式。

若APPLESOFT 有用到ROM ，則設定之（ A5B2 ）。
若成功，則冷起動DOS（9D84 ）。
否則，設定狀態旗號指出INTEGER BASIC 是現用的。
將主檔名緩衝器設定爲" APPLESOFT "。
設定旗號以標示RAM APPLESOFT並進行冷起動。
進入RUN 命令處理常式。

A59E-A5B1 INT 命令處理常式。
若INTEGER BASIC 有用到ROM ，則設定之（ A5B2 ）。
若不成功，則指出" LANGUAGE NOT AVAILA-BLE"（語言未提供）。
否則，清除RUN 被截斷旗號。
冷起動DOS （ 9D84 ）。

A5B2-A5C5 設定所要的BASIC的ROM。
（對APPLESOFT ，傳送$4C 給此常式，對INT-EGER則送$20，因在這些BASIC中，這幾個拜都是在$E000。不管目前的BASIC是那一個，都將有效。）
若所要的BASIC已經可用，則跳出。
嘗試選擇ROM 卡片。
若所要的BASIC現在已可用，則跳出。
嘗試選擇已裝上的ROM 。
若所要的BASIC現在已可用，則跳出。
否則，送出錯誤返回碼並跳出。

A5C6-A5DC EXEC 命令處理常式。

開啓檔案（A2A3）。

將檔案緩衝器位址抄錄到AAB4，AAB5 的EXEC緩衝器指標。

設定旗號，標示EXEC目前在使用中（AAB3）。

進入POSITION命令處理常式以跳過R行。

A5DD-A60D POSITION命令處理常式。

找出開檔緩衝器（A764）。

若未找到，則開啓一個，並設爲主文（TEXT）檔

將緩衝器指標抄錄到檔案處理程式參數表中。

若指令中沒有R，則跳出。

A5F2 否則，測試R值，若爲零則跳出。

將R值減一。

逐個拜的讀取檔案，直到碰到CR（一行結束，$8D）爲止。

若檔案結束，則指出"END OF FILE"（檔案結束）。

否則，進入A5F2跳掉下一個資料錄。

A60E-A625 將一個資料拜寫到檔案上。

確信BASIC正在執行一個程式（A65E）。

若不是，則關閉檔案並暖起動DOS 。

將CSWL的截斷狀態設爲6（跳過催促符號）。

A630 讀取檔案的下一個拜（A68C）。

若檔案未結束，則進入A644 。

否則，關閉檔案。

若狀態不是3（EXEC），則指出"END OF

DATA"（資料結束）。

跳回主叫方。

A644 若資料拜是小寫字母，可將最高比次OFF 掉，則監督程式中的GETIN常式會將此當成大寫處理。

進入DOS 時，先將A暫存器中的資料拜保存。

若需要時，利用行索引，將存在$200（輸入行緩衝器）的資料拜的最高比次ON上，以便呈現小寫。

跳回DOS （9FB3）。

A65E-A678 檢查BASIC 是否正在執行一個程式或是在立即（Immediate）命令模式中。

若現用的BASIC 是 INTEGER，則進入A672。

若行數號碼超過 65280，且催促符號是"]"，則APPLESOFT是在立即模式。

否則，就是正在執行一個程式。

跳回主叫方，並送回相當的返回碼。

A672 檢查$D9決定BASIC 是否正在執行一個程式，跳出時並送出相當的返回碼。

A679-A681 關閉目前的檔案並暖起動DOS。

A682-A68B EXEC從檔案讀取一個拜。

選擇EXEC檔案緩衝器。

將緩衝器的位址抄錄到檔案處理程式的參數表中。

設定狀態爲3（輸入回應）。

進入A62D去讀取一個拜。

A68C-A69C 從主文（TEXT）檔中讀取下一個拜。

於檔案處理程式參數表中設好"讀取一個拜"的參數。

叫用檔案處理程式的推動常式。

跳回主叫方，並傳送回此一資料拜。

A69D-A6A7 設定 $40 ，$41 使指向 EXEC檔的緩衝器。

A6A8-A6C3 檔案處理程式推動常式。

叫用檔案處理程式本身（AB06）。

若無錯誤，則跳回主叫方。

否則，使 $40 ，$41 指向檔案的緩衝器。

若找到該緩衝器，則在其檔案名稱欄放入零使之釋放。

若其錯誤不是" END OF DATA"（資料結束）
，則印出錯誤情報。

否則，假設讀取到$00而跳回主叫方。

A6C4-A6D4 幾種錯誤的指示。

A6C4 " COMMAND SYNTAX ERROR "（命令的句法錯誤）。

A6C8 " NO FILE BUFFERS AVAILABLE "（沒有可用的檔案緩衝器）。

A6CC " PROGRAM TOO LARGE "（程式太大）。

A6D0 " FILE TYPE MISMATCH "（檔案型式不符）。

A6D5-A701 錯誤處理常式。

設定暖起動旗號並清除狀態（BCE6）。

若 APPLESOFT ONERR 目前在使用中，則進入 A6EF 。

否則，印出RETURN BELL RETURN（回車，響鈴，回車）。

印出錯誤指示的文字（A702）。

再送出另一個RETURN（回車）。

A6EF　更換DOS 截斷處理向量。

若有BASIC 的程式正在執行，則傳送錯誤碼給 BASIC的錯誤處理常式。

否則，暖起動DOS 。

A702-A719　印出錯誤指示的文字。

利用錯誤碼爲索引，從指示表（A971）中一個拜一個拜的印出指示文字。

最後一個字母的最高比次設爲 ON。

A71A-A742　完成檔案處理程式的參數表。

抄錄册號數到參數表中。

抄錄推動器號數到參數表中。

抄錄槽路號數到參數表中。

抄錄主檔案名稱緩衝器的位址到參數表中。

將檔案緩衝器的位址存在 $40 ，$41 。

跳回主叫方。

A743-A74D　將主檔名抄錄到檔案緩衝器的檔名欄中。

A741-A75A　將目前的緩衝器指標抄錄到檔案緩衝器的參數表中。

抄錄檔案處理程式的工作區緩衝器指標。

抄錄軌道／段落（T／S）表列段落緩衝器指標。

抄錄資料段落緩衝器的位址。

抄錄下一個檔案緩衝器的鏈結位址。

跳回主叫方。

A75B-A763　將狀態重新設爲 0 ，並設定暖起動旗號。

A764-A791　找出一個開啓或空閒的檔案緩衝器。

將$45清為0，以假定沒有空閒的檔案緩衝器。
將$40，$41指向鏈上的第一個緩衝器。
進入A773。

A76E　將$40，$41指向鏈上的下一個緩衝器。
若鏈已到結尾，則跳出並送出"檔案未開啓"的碼。

A773　取得檔名欄的第一個拜。
若為0（檔案緩衝器為空閒），則將檔案緩衝器位址保存在$44，$45，當作是一個可用的緩衝器，然後進入A76E。
否則，檢查主檔名緩衝器中的名稱是否和此檔案緩衝器中的名稱相符合。
若不符合，則進入A76E取得次一緩衝器。
若符合，則跳回主叫方並指出已找到開啓的檔。

A792-A799　將$40，$41指向鏈上的第一個檔案緩衝器。

A79A-A7A9　將$40，$41指向鏈上的下一個檔案緩衝器。

A7AA-A7AE　取得檔案緩衝器中檔名的第一個拜。

A7AF-A7C3　檢查目前的緩衝器是否屬於EXEC。
EXEC在使用中嗎？
若不是，則跳出。
若是，則目前緩衝器的位址和EXEC的是否符合？
跳回主叫方並送出相當的訊息碼。

A7C4-A7D3　檢查檔案型式。
已開啓的檔案型式是否與所要的檔案型式符合？
若是，則跳出。
否則，將鎖制比次OFF掉，並再測試。
若符合，則跳出。

否則，關閉檔案並指出"檔案型式不符"（FILE TYPE MISMATCH）。

A7D4-A850 重新設定DOS 檔案緩衝器的鏈結。
將$40，$41 指向第一個緩衝器。
設定對MAXFILES值的計數器。

A7E5 將檔名欄中存入零，以標示爲空閒。
在緩衝器中設定鏈結指標，指向檔案處理程式工作區（在檔名欄之前的45 個拜）。
設立指向T／S表段落緩衝器（檔案處理程式工作區緩衝器之前的256個拜）的鏈結指標。
設立指向資料段落緩衝器的鏈結指標。
將計數器遞減。
若結果爲0，則進入A82D 設定HIMEN。
否則，設定指向下一個檔案緩衝器（在資料段落緩衝器之前的38個拜）的鏈結。
進入A7E5 建立下一個緩衝器。

A82D 設定最後一個緩衝器的鏈結爲$0000。
若INTEGER BASIC 爲使用中，則進入A846。
否則，設定APPLESOFT 的HIMEN和字串開始（STRING START）指標於第零頁中，指向最後一個緩衝器的下一個位址。
跳回主叫方。

A846 設定INTEGER BASIC的HIMEN和PROGRM START 指標，以指向最後緩衝器的下一個位址。
跳回主叫方。

A851-A833　更換DOS 鍵盤／視訊的截斷處理向量。
DOS 鍵盤（KSWL）向量仍然設定中嗎？
若是，則進入A86A。
否則，將目前的KSWL 向量（$38，$39）保留在AA55，AA56而用DOS 的截斷處理常式的位址取代。

A86A　DOS 視訊（CSWL）向量仍然設定中嗎？
若是，則跳回主叫方。
否則，將目前的CSWL 向量（$36，$37）保留在AA53，AA54而用DOS 的截斷處理常式的位址取代。
跳回主叫方。

A884-A908　DOS 命令名稱主文表。
這個表包含了每一個DOS 命令的ASCII 名字，按其命令索引值的順序排列。每一命令的最後一字的最高位比次（MSB）均設爲ON。命令的順序如下：
INIT, LOAD, SAVE, RUN, CHAIN, DLLETE, LOCK, UNLOCK, CLOSE, READ, EXEC, WRITE, POSITION, OPEN, APPEND, RENAME, CATALOG, MON, NOMON, RR#, IN#, MAXFILES, FP, INT, BSAVE, BLOAD, BRUN, VERIFY。
假：INIT爲$49, $4E, $49, $D4（INIT）。

A909-A940　命令的有效鍵語表。
這個表是用來決定任一DOS 命令所需要的或可能

賦予的鍵語。

每一命令有兩個拜的述句當作16個旗號，以標示可能賦予的鍵語，其每一個旗號比次的設定情形為：

比次	意　　義
0	檔名有效，但屬於任選的（Optional）
1	命令中沒有與位置有關的運算元。
2	需要檔名＃1。
3	需要檔名＃2
4	需要槽路號數，且為位置有關之運算元
5	需要MAXFILES值，且為位置有關之運算元。
6	命令只可從程式中發出
7	若找不到檔案，則命令可另建新檔
8	C,I,O等鍵語有效
9	V鍵語有效
10	D鍵語有效
11	S鍵語有效
12	L鍵語有效
13	R鍵語有效
14	B鍵語有效
15	A鍵語有效

因此，對一典型命令，OPEN，其述句為$ 2378，即比次2,6,7,9,10,11,12被設定，表示命令有一個檔名運算元，命令只能從程式中發出，可以建立新檔，及V,D,S和L等鍵語有效。

各命令的述句如下：

```
The command entries ar
     INIT        2170
     LOAD        A070
     SAVE        A170
     RUN         A070
     CHAIN       2070
     DELETE      2070
     LOCK        2070
     UNLOCK      2070
     CLOSE       6000
     READ        2206
     EXEC        2074
     WRITE       2206
     POSITION    2204
     OPEN        2378
     APPEND      2270
     RENAME      3070
     CATALOG     4070
     MON         4080
     NOMON       4080
     PR#         0800
     IN#         0800
     MAXFILES    0400
     FP          4070
     INT         4000
     BSAVE       2179
     BLOAD       2071
     BRUN        2071
     VERIFY      2070
```

A941-A94A　鍵語名稱表。

此表包含了所有DOS 鍵語的ASCII 名稱（按標準順序排列）。每一個鍵語名稱佔用一個拜：

V，D，S，L，R，B，A，C，I，O

A94B-A954　鍵語旗號比次的位置表。

此表指出每一鍵語在命令有效鍵語表中第二個拜及指示旗號（ AA65 ）中的比次位置，該指示旗號是用以指出應該在命令中出現的鍵語。

```
V - 40
D - 20
S - 10
L - 08
R - 04
B - 02
A - 01
C - C0 ...
I - A0 ...  } 在有效鍵語表中未用到
O - 90 ...
```

A955-A970　鍵語數值有效範圍表。

此表指出每一鍵語數值可允許的範圍，每一鍵語有四個拜的述句，最大值及最小值各佔兩個拜。其值如下：

KEYWORD	MIN	MAX
V	0	254
D	1	2
S	1	7
L	1	32767
R	0	32767
B	0	32767
A	0	65535

因為C，I，O沒有數目值，故不在表中。

A971-AA3E　錯誤指示主文表。

此表包含了每一錯誤碼的主文（按錯誤碼的號序排列）。

NUMBER	TEXT
0	RETURN BELL RETURN
1	"LANGUAGE NOT AVAILABLE"
2	"RANGE ERROR" (Bad file manager opcode)
3	"RANGE ERROR" (Bad file manager subcode)
4	"WRITE PROTECTED"
5	"END OF DATA"
6	"FILE NOT FOUND"
7	"VOLUME MISMATCH"
8	"I/O ERROR"
9	"DISK FULL"
10	"FILE LOCKED"
11	"SYNTAX ERROR"
12	"NO BUFFERS AVAILABLE"
13	"FILE TYPE MISMATCH"
14	"PROGRAM TOO LARGE"
15	"NOT DIRECT COMMAND"

AA3F-AA4F　錯誤指示主文偏移量索引表。
此表包含了任一錯誤指示與其在上表中主文之間拜的偏移量。
每一錯誤碼號數的進入點佔用一個拜。

AA4F-AA65　DOS 主常式變數群。

AA4F　目前的檔案緩衝器位址（2拜）。

AA51　狀態旗號：$01＝讀取，$00＝暖起動。
$80＝冷起動，$40＝APPLESOFT RAM

AA52　DOS CSWL 截斷狀態號碼。

AA53　實際CSWL 處理常式的位址（2拜）。

AA55　實際KSWL 處理常式的位址（2拜）。

AA57　MAXFILES值。

AA59　進入DOS 時，S，X，Y及A蓄錄器的保存區（4個拜）。

AA5D　命令的行索引值（該行中的偏移量）。

AA5E　MON 旗號（C＝$40，I＝$20，O＝$10）。

AA5F　前一命令的索引乘以2。

AA60　LOAD和BLOAD 的範圍長度（2個拜）。

AA62　若有等待中的命令，此爲其索引。

AA63　臨時變數（計數器，指示索引……等）。

AA64　目前鍵語的索引。

AA65　出現在命令旗號中的鍵語。

AA66-AA74　由命令中剖析而得的或自然設定的鍵語值。

```
AA66 Volume (2 bytes)
AA68 Drive (2 bytes)
AA6A Slot (2 bytes)
AA6C Length (2 bytes)
AA6E Record (2 bytes)
AA70 Byte (2 bytes)
AA72 Address (2 bytes)
AA74 MON value (one byte)
```

AA75-AA92 主檔名緩衝器。

AA93-AAB0 次檔名（RENAME）緩衝器。

AAB1-AAC0 DOS 主常式的常數和變數。

AAB1 MAXFILES的自然值（$03）。

AAB2 Control-D（$84）。

AAB3 EXEC 檔使用中旗號（$00＝未使用中）。

AAB4 EXEC 檔案緩衝器位址（2個拜）。

AAB6 現用的BASIC旗號：$00＝INTEGER，$40＝APPLESOFT ROM，$80＝APPLESOFT RAM

AAB7 "RUN 被截斷"旗號

AAB8 "APPLESOFT"的ASCII碼（9個拜）。

AAC1-AAC8 檔案處理程式的常數。

AAC1 RWTS參數表的位址（B7E8）。

AAC3 VTOC段落緩衝器的位址（B3BB）。

AAC5 目錄表段落緩衝器的位址（B4BB）。

AAC7 DOS 最後一個拜的位置加一的位址（C000）。

AAC9-AAE4 檔案處理程式的常式進入表。

此表包含了14個檔案處理程式的功能處理常式的位址，每一位址佔用2個拜，按其運算碼順序排列。

AAE5-AAF0　檔案處理程式的讀取功能副常式進入表。

此表包含了6個讀取功能副常式的位址，各佔2拜

AAA1-AAFC　檔案處理程式的寫出功能副常式進入表。

此表包含了6個寫出功能副常式的位址，各佔2拜

AAFD-AB05　檔案處理程式外叫程式的進入點（從$3D6跳入)。

X暫存器是否爲0。

若是，則模擬INIT命令索引允許一新檔。

否則，模擬LOAD命令索引要求舊檔。

往下進入檔案處理程式主程式進入點。

AB06-AB1B　檔案處理程式的主進入點。

將S暫存器存在B39B。

將檔檔緩衝器存回檔案處理程式的工作區(AE6A)

確信運算碼不超過13。

若超過，則跳回並送出返回碼＝2（無效運算碼）

利用運算碼爲索引，從檔案處理程式的常式進入表

中找出位址，而經由RTS指令，進入適當的常式

AB1F-AB21　跳回並送出返回碼＝2（運算碼不對）。

AB22-AB27　OPEN功能處理常式。

叫出開檔共用常式（AB28）。

跳出檔案處理程式。

AB28-ABD8　開檔共用常式。

重新設定檔案處理程式工作區的變數自然值(ABDC)。

設定段落長度爲256。

確使資料錄長度不爲0，若爲0，則改爲1。

將資料錄長度存入檔案處理程式的工作區。

找到或配置一個檔案的目錄表進入點（B1C9）

若檔案已經存在，則進入ABA6。

否則，若是正在執行"APPLESOFT"，則設定"語言不能使用"(LANGUAGE NOT AVAILABLE)的返回碼後跳出。

若不是正在執行"APPLESOFT"則設定"檔案找不到"(FILE NOT FOUND)的返回碼後跳出。

AB64　設定目錄表進入點中的段落計數爲1（即在開始時僅有一個T/S表段落。）

配置一個段落給T/S表（B244）。

將此段落的號數存入目錄表進入點及檔案處理程式工作區中的第一個及目前使用的T/S表段落號數中。

將軌道號數也存入上兩個地方。

將所要的檔案型式移到目錄表進入點。

將目錄段落寫回目錄（B037）。

選擇T/S表緩衝器（AF0C）。

將其清爲0（B7D6）。

並寫回去（AF3A）。

設定返回碼爲6（"檔案找不到"）。

ABA6　將T/S表的軌道/段落放入檔案處理程式工作區中第一個T/S表變數中的目錄表進入點中。

將目錄表上的檔案型式抄錄到參數表中，以便傳回主叫方，及檔案處理程式的工作區中。

將檔案中的段落數抄錄到工作區中。

將目錄表到進入點的偏移量保存在工作區中。

設定資料結束指標爲"無限大"。

將一個T/S表段落所代表的資料拜的數目

122*256（30.5K）設定於工作區中。
讀取第一個T／S表段落（AF5E）。

ABDC-AC05 重新設定檔案處理程式的工作區。
將工作區的全部45個拜全清為0。
將册號的互補值保存在工作區中。
將推動器號數保存在工作區中。
將槽路*16值保存在工作區中。
設定軌道號數為$11（目錄軌道）。
跳回主叫方。

AC06-AC39 CLOSE功能處理常式。
必要時，在磁碟上將資料緩衝器設定檢查點（AF1D）。
必要時，將T／S表緩衝器設定檢查點（AF34）。
釋放經配置而未使用到的段落（B2C3）。
若不須再讀取VTOC，則跳出。
否則，再讀取VTOC（AFF7）。
從目錄表中搜尋該檔案的進入點。
求得此進入點的索引。
根據新檔案的長度修正該進入點中的段落數目。
在磁碟上設立目錄表段落的檢查點。
跳出檔案處理程式。

AC3A-AC57 RENAME 功能處理常式。
叫出共用程式來尋找／開啓檔案。
若檔案已鎖制，則跳出並指出"檔案已鎖制"（FILE LOCKED）。
使$42,$43指向新檔名。

將新檔名抄錄到目錄表進入點內。
將目錄表段落寫回磁碟。
跳出檔案處理程式。

AC58-AC69　READ功能處理常式。
保證其次碼（Subcode）不超過5，若超過時，則跳出並送出返回碼＝3。
利用次碼為索引，找出READ次碼處理常式進入表中之對應常式。
進入該次碼處理常式。

AC6A-AC6C　次碼不對時的返回碼＝3。

AC6D-AC6F　指出“檔案已鎖制”（FILE LOCKED）的錯誤。

AC70-AC86　WRITE功能處理常式。
若檔案已被鎖制，則跳出並指出“檔案已鎖制”。
保證其次碼不超過5，若超過時，則跳出並送出返回碼＝3。
利用次碼為索引，找出WRITE次碼處理常式進入表中之對應常式。
進入該次碼處理常式。

AC87-AC89　“定位並讀取一個拜”次碼處理常式。
叫用定位常式。
進入下面的次碼處理常式。

AC8A-AC92　“讀取一個拜”次碼處理常式。
讀取檔案的下一個拜（ACA8）。
將其存入參數表中，以待傳回主叫方。
跳出檔案處理程式。

AC93-AC95　“定位並讀取一段拜”次碼處理常式。

叫用定位常式。

進入下面的次碼處理常式。

AC96-ACA7 "讀取一段拜"次碼處理常式。

將長度遞減並予檢查(B1B5)。

讀取一個拜(ACA8)。

將$42,$43指向範圍值的位址上,並將位址加1。

將所讀取的拜存入該位址。

回到AC96再做(當長度爲0時跳出檔案處理程式)

ACA8-ACBA 讀取一個資料拜。

必要時,讀取下一個資料段落(B0B6)。

若檔案完畢,則跳出並指出"資料完畢"(END OF DATA)。

否則,從資料段落緩衝器載入資料拜。

將資料錄號數/拜偏移量予以遞增(B15B)。

將檔案位置偏移量予以遞增(B194)。

跳回時並送回所讀取的資料拜。

ACBB-ACBD "定位並寫入一個拜"次碼處理常式。

叫用定位常式。

進入下面的次碼處理常式。

ACBE-ACC6 "寫入一個拜"次碼處理常式。

找到所要寫入的資料拜。

將其寫入檔案(ACDA)。

跳出檔案處理程式。

ACC7-ACC9 "定位及寫入一段拜"次碼處理常式。

叫用定位常式。

進入下面的次碼處理常式。

ACCA-ACD9 "寫入一段拜"次碼處理常式。
抄錄並推進範圍值的位址指標。
取得所要寫入的下一個資料拜。
將其寫入檔案（ACDA）。
測試並遞減範圍長度（B1B5）。
再回到ACCA去做。

ACDA-ACEE 寫入一個資料拜。
必要時，讀取恰當的資料段落（B0B6）。
將所要寫入的資料拜存入段落緩衝器內。
將該資料段落緩衝器設立旗號，以標示要求重寫。
將資料錄號數／拜偏移量予以遞加（B15B）。
經由檔案位置偏移量遞加常式跳出（B194）。

ACEF-ACA5 LOCK功能處理常式。
設定面罩拜爲$80（鎖制）。
進入共用程式（ACFB）。

ACF6-ACFA UNLOCK 功能處理常式。
設定面罩拜爲$00（解制）
進入下面的共用常式（ACFB）。

ACFB-AD11 LOCK/UNLOCK共用常式。
定位／開啓檔案（AF28）。
取得目錄表中進入點的索引。
更正檔案型式拜爲鎖制（$8x）或解制（$0x）。
將目錄表段落寫回磁碟。
跳出檔案處理程式。

AD12-AD17 POSITION功能處理常式。
叫用定位常式。

跳出檔案處理程式。

AD18-AD2A　VERIFY 功能處理常式。

定位／開啓檔案（AB28）。

AD1B　讀取次一資料段落。

若檔案結束，則跳出檔案處理程式。

否則，將段落的位置遞加。

回到 AD1B 再做。

AD2B-AD88　DELETE功能處理常式。

定位／開啓檔案（AB28）。

利用目錄表索引，決定檔案是否已鎖制。

若鎖制，則跳出並指出"檔案已鎖制"。

將目錄表上T／S表段落的軌道號數抄錄到工作區及目錄表進入點本身檔名的最後一字之後。

在目錄表進入點中的T／S表段落軌道號數上存入$FF 以此標示此檔已被刪除。

將T／S表段落的段落號數抄錄到工作區。

將目錄表段落寫回磁碟。

AD54　讀取下一個T／S表段落（AF5E）。

若已結束，則寫入VTOC並跳出檔案管理程式。

否則，選擇T／S表緩衝器（AF0C）。

將索引指向第一個T／S對。

AD5E　若軌道號數爲0或負，則略去。

否則，修改VTOC的比次對映表以釋放資料段落（AD89）。

將索引指向下一個T／S對。

若還有，則進入AD5E。

根據此T／S表段落找出下一個T／S表段落的軌道／段落。
釋放此一T／S表段落（AD89）
若還有下一個，則繼續處理（進入AD54）
否則，寫入VTOC並跳出檔案處理程式。

AD89-AD97 釋放一個段落。
叫用B2DD經由VTOC的比次對映表以收回（deall-ocate）段落。
將工作區的段落配置區清爲0。
跳回主叫方。

AD98-AE2E CATALOG功能處理常式。
設定檔案處理程式工作區的起始初值（ABDC）。
設定V值爲0（補數爲$FF）。
讀取VTOC段落（AFF7）。
在等待鍵入之前，先設好可數22行的計數器。
在顯示幕上，跳過二行。
印出"DISK VOLUME"（磁碟册號）。
將册號經轉換後顯示出（AE42）。
再跳過兩行。

ADCA 讀取下一個目錄表段落。
若已讀完，則跳出檔案處理程式。
設定索引，指向第一個進入點。

ADD1 取得軌道號數。
若爲0，則跳出檔案處理程式。
若爲負，則略過該進入點（檔案已刪除）。
若檔案被鎖制，則顯示出"＊"（可由檢查檔案型

式拜而知）。

利用檔案型式爲索引，從B3A7 處的檔案型式名稱表中找出並印出所找到的一個字母。

印出一個空白。

轉換並印出檔中段落的數目。

印出一個空白。

利用索引找到檔名。

印出檔名。

跳到下一行。

將索引推進到下一個目錄表進入點。

若還有，則進入ADD1 。

否則，進入ADCA讀取下一個目錄表段落。

完成後跳出。

AE2F-AE41　在印出CATALOG 時跳過一行。

送出一個"回車"（CR）信號。

將行計數器遞減。

若仍不爲0，則跳出。

否則，等待鍵入。

然後，重新設定計數器爲21 行。

跳回主叫方。

AE42-AE69　將存在 $44 中的數目轉換爲三位數字字母後印出。

AE6A-AE7D　將檔案緩衝區存回檔案處理程式的工作區。

選擇檔案管理程式工作區的緩衝器。

事先設定參數表中的返回碼＝ 0 。

將檔案緩衝器中檔案處理程式工作區存入的45個拜抄錄到現用的檔案處理程式工作區中。

跳回主叫方。

AE7E-AE8D 將檔案處理程式工作區保存在檔案緩衝器內。

選擇檔案處理程式工作區緩衝器。

將工作區的45個拜抄錄到檔案緩衝區。

跳回主叫方。

AE8E-AF07 INIT功能處理常式。

設定檔案處理程式工作區的起始初值（ABDC）。

叫用RWTS 將磁碟格式化（B058）。

將V值抄錄到VTOC緩衝器。

將可用以進行配置工作的起始軌道設定為$11。

配置方向定為$01（向前）。

將VTOC比次對映表清為0（表示所有段落皆為使用中）。

略過前三個軌道及軌道$11，將B3A0的四個面罩拜抄錄到VTOC比次對映表各軌道的進入點中以釋放軌道，這樣可使前三個軌道及目錄軌道仍標示為使用中。

將目錄表段落緩衝器清為0。

指向目錄表段落緩衝器。

將軌道$11設定在RWTS參數表中。

設好此目錄表段落與下一個段落（軌道$11，段落1）間的鏈結。

叫用RWTS以寫入目錄表段落。

除段落0外，軌道中的其它段落都同此方法寫入。

將最後段落（段落1）的鏈結指標設為0。

將RWTS的參數指向DOS 的載入點（B7C2）。

將DOS 的映像寫入軌道0－2（B74A）。
跳出檔案處理程式。

AF08-AF1C 設定$42,$43以指向欲選擇的緩衝器。

AF08 選取檔案緩衝器中檔案處理程式的工作區緩衝器。

AF0C 選取檔案緩衝器中的T／S表段落緩衝器。

AF10 選取檔案緩衝器中的資料段落緩衝器。

AF1D-AF33 將資料段落緩衝器寫入磁碟，並設定檢查點。
測試旗號，以檢查從上次的讀／寫之後緩衝器是否有改變。
若沒有，則跳回主叫方。
否則，設好RWTS的指標（AFE4）。
叫用RWTS將段落寫入。
將旗號重置，以標示資料段落不需再用檢查點。
跳回主叫方。

AF34-AF4A 將T／S表段落緩衝器寫入磁碟，並設立檢查點。
測試旗號，以檢查從上次的讀／寫之後緩衝器是否會改變。
若沒有，則跳回主叫方。
否則，設好RWTS的指標（AF4B）。
叫用RWTS將段落寫入。
將旗號重置，以標示T／S表段落不需再用檢查點。
跳回主叫方。

AF4B-AF5D 準備一個T／S表段落以供叫用RWTS。
將T／S 表緩衝器的位址抄錄到RWTS參數表中。
取得段落的軌道／段落號數。

跳回主叫方。

AF5E-AFDB 讀取一個T/S表段落到檔案緩衝器。

(在進入時，設定進位(CARRY)旗號以標示此檔案的第一個T/S表是所需要的(C=0)或下一個才是所需要的(C=1))。

保存進入時的進位旗號。

必要時，將目前用到的T/S表段落設立檢查點。

設好參數，準備叫用RWTS(AF4B)。

選擇T/S 表緩衝器(AF0C)。

需要第一個或下一個段落?

若要第一個，則進入AFB5繼續。

否則，從這一個取得下一個T/S表的鏈結。

若鏈結非0，則可用來找到下一個段落，並進入AFB5。

否則，必然不在此檔案的T/S表中。

若正在讀取檔案，則跳出時並送出錯誤指示碼。

否則，配置一個新的段落(B244)。

將表的T/S表段落指向該新的軌道/段落。

並將老的T/S表段落寫回磁碟。

將該緩衝器清爲0，變成新的T/S表段落。

計算此段落相對於第一個表列段落(存在此段落的+5,+6位置上)的數值，並存入緩衝器。

設定RWTS 運算碼，以寫入新的T/S表段落到磁碟。

AFB5 設定RWTS 運算碼，以讀取老的T/S表(除非剛配置好)。

設定軌道和段落，並叫用RWTS 來讀取老的表或寫入新的表。

計算此表最後一個段落（加一）的相對段落號數，並存入工作區中。

跳回主叫方並送出正常返回碼。

AFDC-AFE3 讀取一個資料段落。

叫出AFE4，設好叫用RWTS 的參數。

設定RWTS 讀取（READ）運算碼，並進入RWTS 推動常式去執行。

AFE4-AFF6 準備RWTS 所需的資料段落。

將資料段落緩衝器的位址抄錄到RWTS 參數表。

取得其軌道/段落。

跳回主叫方。

AFF7-B010 讀／寫VTOC緩衝器。

AFF7 讀取VTOC進入點，進入AFFD。

AFFB 寫入VTOC進入點，往下進行。

AFFD 共用常式。

將VTOC段落緩衝器位址抄錄到RWTS 參數表中取得其軌道號數並使用段落$00 。

經由RWTS 推動常式跳出。

B011-B036 讀取目錄表段落。

（進入時若進位旗號爲0，則讀取第一個目錄表段落，若爲1，則讀取下一個。）

保存進入時的旗號。

設定緩衝器指標（B045）。

要第一個段落或下一個？

若要第一個，則從VTOC的+1，+2 偏移位址上取得目錄表段落的軌道／段落號數。

否則，從目錄表段落的+1，+2 偏移位址上取得軌道／段落號數。若其軌道號數爲 0 ，則跳出並送出錯誤指示碼（目錄表結束）。

叫用RWTS 以讀取段落。

跳出並送出正常返回碼。

B037-B044 寫入目錄表段落。

設定緩衝器指標。

從工作區中找出其軌道／段落號數。

叫用RWTS 予以寫入後跳出。

B045-B051 準備RWTS 需要的目錄表緩衝器。

將目錄表緩衝器的位址抄錄到RWTS 參數表中。

跳回主叫方。

B052-B0B5 讀／寫，軌道／段落（ RWTS ）推動常式。

將軌道／段落號數設定在RWTS 參數表中。

B058 設定命令碼（讀、寫…等）。

若要寫，則設定旗號（ B5D5 ）。

設定所要的册號於參數表中。

設定槽路 *16的値於參數表中。

設定推動器號數於參數表中。

設定段落大小於參數表中。

設定IOB 型式於參數表中。（ $01 ）。

叫用RWTS ，傳送參數表指標。

將找到的實際册抄錄到檔案處理程式的參數表中。

將RWTS 參數表中希望的册號予以重置。
若沒有任何錯誤，則跳回主叫方。
否則，取得返回碼。
將"册號不符"的返回碼定爲7，"寫入保護"的返回碼定爲4，所有其它的錯誤定爲8（I／0錯誤)。
跳出檔案處理程式。

B0B6-B133 必要時，讀取下一個資料段落。
目前的檔案位置是否在記憶體中的資料段落中？
若是，則進入B12C。
否則，在資料段落緩衝器上設立檢查點。
目前的檔案位置是否在此T／S表範圍的前後？
若不是，則進入B0F3。
否則，讀取此檔案的每一個T／S表，從第一個開始，到找到符合的一個爲止。
若都找不到，則跳出並送出錯誤指示（檔案結束，仍未讀取到）。

B0F3 資料在此一T／S表段落中。
計算對此T／S表段落中爲符合的進入點的位移量。
選擇T／S表緩衝器。
取得所要的資料段落的軌道號數。
若不爲0，則進入B114。
否則，若不是要寫入，則跳出並指出錯誤（讀不到資料）。
若要寫入，則配置一個新段落並將其軌道／段落在此表列中的位置存入（B134）。
進入B120。

B114	利用T／S表進入點中找到的軌道／段落來讀取老的資料段落。
B120	將前次讀取的段落數存入工作區中。
B12C	選擇資料緩衝器。 取得偏移的拜數，並正常的跳回主叫方。
B134-B15A	在檔案中加入一個新的資料段落。 配置一個段落（B244）。 將軌道／段落號數存入T／S表進入點中。 選擇資料緩衝器並清為0。 設定旗號，以標示T／S表段落和資料段落緩衝器需要設立檢查點。 跳回主叫方。
B15B-B193	將資料錄號數和對檔案的偏移拜數予以遞加。 將目前的資料錄號數和偏移拜數抄錄到檔案處理程式的參數表中，以便傳回主叫方。 將工作區中的偏移拜數予以遞加。 若偏移的拜數等於資料錄長度，設設定偏移拜數為0，並將資料錄長度遞增。 跳回主叫方。
B194-B1A1	將檔案位置偏移量增加。 將目前使用中的段落的拜偏移量遞增。 若段落結束，則將段落號數加一。 跳回主叫方。
B1A2-B1B4	抄錄並推進範圍位址。範圍位址從檔案處理程式參數表中抄錄到$42。

將參數表中的範圍位址予以遞加。
跳回主叫方。

B1B5-B1C8 將範圍長度遞減。
將檔案管理程式參數表中的範圍長度減一。
若其值爲0，則跳出檔案處理程式。
否則，跳回主叫方。

B1C9-B21B 在目錄中找出或配置一個目錄表進入點。
讀取VTOC段落（AFF7）。
將$42,$43指向所要找出的檔案名稱。
設定行程爲1（找出檔案）。

B1D8 重新設定目錄表的段落偏移量（第一個段落）。

B1E1 將段落偏移量予以遞加。
讀取目錄表段落。
若目錄表結束，則進入B23A。
設定對第一個檔案進入點的進入點索引。

B1EB 取得軌道號數。
若爲已被刪除，則略過進入點，進入B217。
若爲空的，表示目錄表結束，則進入B212。
將索引推進以尋找目錄表中的檔名。
與所要的檔名比較。
若符合，則送回進入點索引並跳出。

B20B 若不符，則將索引推進以指向段落中的下一個進入點，並回到B1EB再做。
若段落結束，則進入B1E1讀取下一個段落。

B212 若行程爲1，則進入B1D8開始第二行程。

B217 若行程爲1，則進入B20B略過進入點。

若為行程 2 ，則往下執行以配置進入點。

B21C-B22F 將檔名抄錄到目錄表進入點中。

將索引往下找出目錄表進入點中的檔名欄。

抄錄30個拜的檔名到目錄表進入點。

重新存回目錄表索引，並跳回主叫方。

B230-B239 將索引推進，以指向段落中下一個目錄索引。

將索引加35（進入點的長度）。

測試段落是否結束，並跳回主叫方。

B23A-B243 將目錄表的搜尋工作轉換成行程 2 。

若為行程 1 ，則轉換成行程 2，並進入B1D8 。

若為行程 2 ，則跳出檔案處理程式並指出"磁碟已滿"（DISK FULL ）的錯誤。

B244-B2C2 配置一個磁碟段落。

目前是否有一個軌道配置給此檔？

若無，則進入B 26A以找出一個含有空閒段落的軌道。

B249 否則，將段落號數遞減以取得下一個可能為空閒的段落號數。

若此軌道上已無可用的段落，則進入B265 ，以尋找新的軌道。

否則，將軌道比次面罩旋轉一位，以取得此段落的對映比次。

若此段落在使用中，則跳回B249再做。

否則，將檔案的段落計數加一。

傳回段落號數（軌道號數在B5F1 ）。

並跳回主叫方。

B265 標示目前沒有軌道在使用中。

B26A　清除配置旗號，以便對所有的軌道做一次澈底搜尋，將空閒可用的段落找出。

讀取VTOC段落。

B272　取得前次配置的軌道，加上方向值，則得出下一個要檢查的軌道（+1 或 -1）。

是否又回到軌道 0 ？

若是，則進入B284 。

否則，是否超過了軌道 34 ？

若是，則將方向相反後進入 B28E 。

B284　是否第二次來到軌道 0（檢查配置旗號）？

若是，則跳出並指出"磁碟已滿"的錯誤。

否則，設定配置旗號作爲標記。

設定方向爲向前（+1 ）

B28E　從目錄表軌道（ 17+或-1 ）開始。

計算比次對映表索引（軌道號數 *4 ）。

將VTOC的比次對映表抄到工作區，並檢查四個拜是否全部爲 0（表示軌道已滿）。

之後，先將VTOC中的四個拜皆設爲 0（配置所有的段落）。

若軌道中無空閒的段落，則跳到B 272 再試下一個軌道。

否則，將VTOC寫回磁碟，以確保檔案的完整性。

將段落號數設定爲軌道中的最後一個段落。

跳到B 249 配置一個空的段落給此檔案。

B2C3-C2DC　釋放現用軌道中已配置的段落，並設定VTOC的檢查點。

是否已配置一個軌道給此檔案？
若沒有，則跳回主叫方。
否則，讀取VTOC。
取得下一個可能被使用的段落（在配置過程中，軌道對映移動的次數可表示段落號數。）
叫用B2DD將軌道比次對映表移位還原，並合併到VTOC比次對映表內。
標示無軌道被配置。
跳回主叫方。

B2DD-B2FF 將檔案管理程式配置區中的面罩拜移位後，回來與VTOC的比次對映表運算，以釋放段落。
（若進位旗號有設定，則目前使用的段落亦被釋放。）
將四個拜的軌道比次對映全部轉動一次。
重複其次數使與所欲配置的段落數一樣。
計算此軌道的對映表在VTOC中的索引。
若為0，則跳出。
將檔案管理程式的對映比次與VTOC中的對映比次合併（"或"運算）以釋放此檔案未曾使用到的段落。
跳回主叫方。

B300-B35E 計算檔案位置。
設定傳交給檔案管理程式工作區參考數表中的資料錄數及其段落的偏移量。
清除段落偏移量的高位部份。
執行16個拜的乘法運算如下：

檔案位置（3個拜）＝資料錄數目＊資料錄長度
加入從參數表到此檔案位置（以B5E4，B5E5，B5E6，中所存的三個拜來表示）的拜偏移量。
跳回主叫方。

B35F-B37D 錯誤時的跳出。

B35F 返回碼＝1，指示"語言不可使用"（LANGUAGE NOT AVAILABLE）。

B363 返回碼＝2，指示"範圍錯誤"（RANGE ERROR）（運算碼錯誤）。

B367 返回碼＝3，指示"範圍錯誤"（運算次碼錯誤）

B368 返回碼＝4，指示"寫入保護"。

B36F 返回確＝5，指示"資料完畢"。

B373 返回碼＝6，指示"找不到檔案"。

B377 返回碼＝9，指示"磁碟已滿"（所有的檔案皆關閉）。

B37B 返回碼＝A，指示"檔案已鎖制"。

B37F-B396 跳出檔案管理程式。

B37F 沒有錯誤時的跳出。
將返回碼設定爲0。
清除進位旗號並進入B386。

B385 設定進位旗號以指示錯誤.

B386 將返回碼存入參數表中。
在RWTS 可能變動監督程式狀態暫存器（$48）後，予以清除。
將檔案處理程式的工作區抄錄到檔案緩衝器（AE7E）。

存回狀態及堆疊暫存器的原值。
跳回檔案處理程式的主叫方。

B397-B3A3 檔案處理程式臨時使用的地方。

B397 目前目錄表段落的軌道／段落號數（2個拜）。

B39B S暫存器的保存區。

B39C 目錄表索引。

B39D 目錄的行計數器／目錄表查表旗號／等。

B39E 鎖制／解制 面罩／配置旗號／等。

B3A0 供INIT用的四個面罩拜，用來改變VTOC中的比次對映表以釋放軌道。

B3A4-B3A6 十進位轉換表（1，10，100）。

B3A7-B3AE 供CATALOG 使用的檔案型式及名稱表。
檔案型式為：T，I，A，B，S，R，A，B對應的十六進位值分別為：$00,$01,$02,$04,$08,$10,$20 $40。

B3AF-B3BA ASCII 文字"DISK VOLUME "以相反順利存放，供CATALOG用。

B3BB-B4BA VTOC段落緩衝器。

B3BC 第一個目錄表段落的軌道／段落。

B3BE DOS 發行號碼（1，2或3）。

B3C1 磁碟的册號。

B3E2 在每一T／S表段落中的進入點個數。

B3EB 下一個要配置的軌道。

B3EC 軌道配置的方向（+1 或 -1）。

B3EF 磁碟上的軌道數目。

B3F0 磁碟上的段落號數。

B3F1　段落上的拜個數（以2個拜表示）。

B3F3　軌道0的比次對映。

B3F7　軌道1的比次對映。

　　　等等。

B47B　軌道34的比次對映。

B4BB-B5BA　目錄表段落的緩衝器。

B4BC　下一個目錄表段落的軌道／段落號數。

B4C6　第一個目錄表進入點和T／S表的軌道號數。

B4C7　T／S表的段落號數。

B4C8　檔案型式及鎖制比次。

B4C9　檔名欄（30個拜）。

B4E7　檔案的段落數（包含T／S表）。

B5BB-B5D0　檔案處理程式的參數表。

B5BB　運算碼。

B5BC　次碼。

B5BD　八個與運算碼有相關性的變數拜。

B5C5　返回碼。

B5C7　檔案處理程式工作區緩衝器的位址。

B5C9　T／S表段落緩衝器的位址。

B5CB　資料段落緩衝器的位址。

B5CD　鏈上的下一個DOS 緩衝器位址（未使用）。

B5D1-B5FD　檔案處理程式的工作區。

B5D1　第一個T／S表段落的軌道／段落號數。

B5D3　目前的T／S表段落的軌道／段落號數。

B5D5　各旗號：80＝T／S表須要設檢查點。

　　　　　　40＝資料段落需要設檢查點。

20＝VTOC段落須要設檢查點。

02＝前一個運算是"寫入"。

B5D6 目前的資料段落的軌道／段落號數。

B5D8 用以找出檔案進入點的目錄表段落索引。

B5D9 在目錄表段落中對檔案目錄表進入點的索引。

B5DA 一個T／S表指出的段落數目。

B5DC 表中第一個段落的相對段落號數。

B5DE 表中最後一個段落的相對段落號數加1。

B5E0 最後一次讀取到的段落相對段落號數。

B5E2 段落長度的拜數。

B5E4 檔案位置（3個拜），段落偏移量，在段落中的拜偏移量。

B5E8 由OPEN而得知的資料錄長度。

B5EA 資料錄號數。

B5EC 在資料錄中的拜偏移量。

B5EE 檔案中的段落數目。

B5F0 段落配置區（6個拜）。

欲配置的下一個段落（以移位的次數表示）。

目前配置的軌道。

目前配置中的軌道的4個比次對映拜，轉動移位到下一個要配置的段落。

B5F6 檔案型式。

B5F7 槽路號數乘以16。

B5F8 推動器號數。

B5F9 册號（補數）。

B5FA 軌道號數。

B5FE-B5FF　未使用。

B600-B6FF　Boot 2/RWTS映像由此開始。

B600　Boot1的映像，可以寫入經過INIT處理過的磁碟的軌道0，段落0中。

B65D　DOS3.3程式修補區。

B65D　APPEND 程式修補旗號。

B65E　APPEND 程式修補。當檔案處理程式的推動常式碰到一個不是資料結束的錯誤時就跳到此處。
找出並釋放檔案緩衝器。
清除APPEND 旗號。
取得錯誤號碼並將錯誤印出（A6D2）。

B671　APPEND 程式修補。由APPEND 命令處理常式跳出來此，若APPEND 旗號有設定時則將資料錄號數遞增，清除旗號。經由POSITION而跳出。

B686　VERIFY 程式修補。由"輸出/輸入一段拜"常式跳來此，而在執行了SAVE或BSAVE之後，叫用VERIFY而跳出。

B692　APPEND 程式修補。若檔案處理程式的推動常式返回碼是"END OF DATA"時跳來此。
測試檔案位置是否為0。
若不為0，則設定APPEND（附加）旗號，並跳回主叫方。
若為0（在檔案開頭），則將資料錄號數和偏拜移量抄錄到檔案處理程式的參數表中並送回一個零資料拜給主叫方。

B6FE　Boot2第一頁的頁位址。

B6FF	Boot2的段落數目（頁數）。
B700-B749	DOS 第二階段靴式載入程式（Boot Loader）。 設定RWTS 參數表，以便從磁碟讀取DOS。 叫用"讀／寫數個記憶頁"常式（$B793）。 建立新的堆疊。 叫用SETVID($FE93)和SETKBD($FE89)。 跳出到DOS 冷起動常式（$9D84）。
B74A-B78C	將DOS 放進軌道0－2。 設定RWTS 參數，以便將DOS 寫入磁碟。 叫用"讀／寫數個記憶頁"常式（$B793）。 跳回到主叫方。
B78D-B792	未使用。
B793-B7B4	讀／寫數個記憶頁。 經由外在進入點叫用RWTS （$B7B5）。 跳回主叫方。
B7B5-B7C1	停止中斷機能並叫用RWTS 。
B7C2-B7D5	設定RWTS 參數以便將DOS 寫入。
B7D6-B7DE	將目前的緩衝器清爲0。 將$42,$43所指的256個拜清爲0。 跳回主叫方。
B7DF-B7E7	DOS第二階段靴式載入程式的參數表。
B7DF	未使用。
B7E0	第二階段DOS 載入的頁數。
B7E1	讀／寫的段落數。
B7E2	第一階段DOS 載入的頁數。

B7E3　INIT DOS 的頁數計次器。
B7E4　RWTS 參數表的指標（2個拜）。
B7E6　第一個階段起動程式位置的指標（2個拜）。
B7E8-B7F8　RWTS 參數表。
B7E8　表的型式。必須是$01。
B7E9　槽路號數乘以16。
B7EA　推動器號數（$01或$02）。
B7EB　所希望册號（0可和任意册號符合）。
B7EC　軌道號數。
B7ED　段落號數。
B7EE　裝置特性表指標（2個拜）。
B7F0　供使用者讀/寫之用的資料緩衝器指標（2個拜）
B7F2　未使用。
B7F3　對零碎段落（256以$00表示）的拜計數。
B7F4　命令碼：0＝尋找，1＝讀取，2＝寫入，4＝定格式。
B7F5　錯誤碼：（進位旗號設定時）$10＝寫入保護，$20＝册號不符，$40＝推動器錯誤，$08＝INIT錯誤。
B7F6　找到的册號數。
B7F7　找到的槽路號數。
B7F8　找到的推動器號數。
B7F9-B7FA　未使用。
B7FB-B7FE　裝置特性表（DCT）。
B7FB　裝置型式（應爲$00）。
B7FC　每軌道的相位（應爲$01）。

D7FD 馬達的時計（2個拜一應爲$EF，$D8）。

B7FF 未使用。

B800-B829 PRENIBBLE（預分塊）常式。

將256個拜（8比位）轉換成342個分塊（nibble），其形式爲00xxxxxx。

將所要轉換的頁數指標存入$3E，$3F中。

資料存放在第一和第二緩衝器中。

在進入時：$3E，$3F中存有資料的指標。

在跳出時：A暫存器的值不定

X暫存器中爲$FF。

Y暫存器中爲$FF。

進位旗號設定。

跳回主叫方。

B82A-B8B7 WRITE常式。

將經過分塊好的資料從第一和第二緩衝器寫入磁碟。

叫用"寫入一個拜"副常式。

寫入5個自動同步拜，資料開始記號（$D5/$AA/$AD），342個資料拜，一個檢核拜，及資料結束記號（$D5/$AA/$EB）。

使用寫入轉換表（$BA29）。

在進入時：X暫存器中爲槽路號數乘以16的值。

在跳出時：若有錯誤，則進位旗號爲設定

若無錯誤，則

A暫存器中的值不定

X暫存器中的值不變

Y暫存器中爲$00。

進位旗號未設定
使用到的位址有$26,$27,$678。
跳回主叫方。

B8B8-B8C1 "寫入一個"拜副常式。
此為時際關係極為嚴格的一段程式。在32周的時間裏寫入一個拜。
跳回主叫方。

B8C2-B8D8 POST NIBBLE常式。
將342個形如00xxxxxx的分塊(nibble)(6比次)轉換成256個拜(8比次)。
分塊存放在第一及第二緩衝器。
資料頁的指標存放在$3E,$3F中。
有進入時:X暫存器中為槽路號數乘以16。
$36,$37為使用者資料指標
$26:第二緩衝器中的拜計數值
進位旗號為設定
跳回主叫方。

B8DC-B943 READ常式。
從磁碟讀取一個段落的資料,並存ˀ ,一及第二緩衝器內(先使用第二緩衝器,由高位到低位,然後使用第一緩衝器,由低位到高位)。
在進入時:X暫存器中為槽路號數乘以16
設定為讀取模式(Q6L,Q7L)
在跳出時:若有錯誤,則進位旗號為設定
若無錯誤,則:
A暫存器中為$AA。

X暫存器不變

Y暫存器：$00

進位旗號被清除

使用了$26

跳回主叫方。

B944-B99F RDADR常式。

讀入一個位址欄。

讀入位址開始記號（$D5/$AA/$96），位址情報（册／軌道／段落／檢核數），及位址結束記號（$DE/$AA）。

在進入時：X暫存器中爲槽路號數乘以16。

設定爲讀取模式（Q6L，Q7L）。

在跳出時：若有錯誤，則進位旗號爲設定

若無錯誤，則：

A暫存器中爲$AA。

X暫存器不變

Y暫存器中爲$00

進位旗號爲清除

$2F：找到的册號

$2E：找到的軌道號數

$2D：找到的段落號數

$2C：找到的檢核數

使用到$26，$27。

跳回主叫方。

B9A0-B9FF SEEKABS 常式。

將磁碟的探臂移動到所要的軌道上。

叫用探臂移動延遲副常式（$BA00）。

在進入時：X暫存器中爲槽路號數乘以16。

A暫存器中爲所要的軌道（在單相磁碟上爲半軌）。

$478：目前的軌道。

在跳出時：A暫存器不定

X暫存器不變

Y暫存器不定

$2A和$478：最終的軌道

$27：最初的軌道（若須要搜尋時）。

使用到$26，$27，$2A，$2B等。

跳回主叫方。

BA00-BA10　探臂移動延遲副常式。

以100毫秒（Usec）爲延遲的時間單位。

在進入時：A暫存器中爲時間單位的次數。

$46，$47：存有裝置特性表的馬達開啓時間的計次（$EF，$D8）

$478：目前的軌道。

在跳出時：A暫存器中爲$00。

X暫存器中爲$00。

Y暫存器不變

進位旗號爲設定

跳回主叫方。

BA11-BA28　探臂移動延遲表。

包含以100毫秒爲單位的時間次數，用以控制步進

達的相位開啓或關閉。

BA29-BA68 寫入轉換表。

拜轉換成分塊（6比次）的對照表。

其値在$96起到$FF之間。

對於拜中有一對以上相鄰的0或沒有相鄰的1者，不包括在內。

BA69-BA95 未使用。

BA96-BAFF 讀取轉換表。

6比次的分塊轉換成8比次的拜的轉換表。

其値在$96起到$FF之間。

對於拜中有一對以上相鄰的0或沒有相鄰的1者，不包括在內。

BB00-BBFF 第一緩衝器。

BC00-BC55 第二緩衝器。

BC56-BCC3 在初始設定時寫入位址欄。

叫用"寫入兩個拜"副常式。

寫入自動同步拜（其個數由Y暫存器中指定），位址開始記號（$D5/$AA/$96），位址情報（册/軌道/段落/檢核數），位址結束記號（$DE/$AA/$EB）。

在進入時：X暫存器中爲槽路號數乘以16

Y暫存器中爲欲寫入的自動同步拜個數

$3E中爲$AA。

$3F中爲段落號數

$41 中爲册號數

$44 中爲軌道號數

在跳出時：A暫存器不定
X暫存器不變
Y暫存器中為$00
進位旗號設定
跳回主叫方。

BCC4-BCDE　"寫入兩個拜"副常式。
此段程式對時際關係很嚴格，將位置情報組碼成偶和奇比次，並在32週的時間內寫入。
跳回主叫方。

BCDF-BCFF　未使用。

BD00-BD18　進入RWTS 的主進入點。
在進入時，將Y及A暫存器分別存入$48,$49，當作IOB 的指標。
設定重調整次數最多為1，搜尋次數最多為4。
檢查槽路號數是否改變，若沒，則進入SAMESLOT($BD34)。

BD19-BD33　修改IOB 中的槽路號數，並等待原來的推動器關掉。

BD34-BD53　SAMESLOT
進入讀取模式，並在讀取時加入延遲，以檢查磁碟是否在轉動。
保存測試結果，且將馬達開啓以防萬一。

BD54-BD73　將IOB 中的指標移到第零頁中，以便以後使用。
將裝置特性表的指標存入$3C，$3D，資料緩衝器指標存入$3E，$3F。
在$47位置中存入裝置特性表的$D8(馬達開啓時間

	)。
	檢查推動器號數是否改變。若沒變，則跳往$BD74
BD74-BD8F	選定適用的推動器，並以$35的高比次來保存在使用中的推動器。1＝推動器1，0＝推動器2。
	取得測試結果。若推動器是開動的，則跳到$BD90。
	利用MSWAIT副常式($BA00)以待電容器放電。
BD90-BDAA	取得目標軌道並叫用MYSEEK副常式(BE5A)以進入。
	再檢查測試結果，若推動器是開動的，則跳到$ $BDAB。
	延遲一段時間以等待馬達速度上升。
BDAB-BDBB	TRYTRK
	取得命令碼。
	若無，則叫用ALLDONE(BE46)將推動器關掉後跳出。
	若為4，則跳到FORMDSK(BEOD)。
	否則，將低位比次移進進位旗號中(設定＝讀取，清除＝寫入)並將其值保存在狀態暫存器。
	若要寫入，則資料先經叫用PRENIB16($B800)予以拆組。
BDBC-BDEC	重新設定"再試"次數最大為48，並叫用RDADR16($B944)以讀取一個位址欄。
	若讀取良好，則跳到RDRIGHT($BDED)。
	若讀取不良，則將再試次數遞減，若不為0，則再試，否則，須準備重新調整。

將重調整次數遞減，若爲 0 ，則叫用DRVERR ($BE04) 指出推動器錯誤。

若不爲 0 ，則重新設定"再尋找"次數爲 4 ，並重新調整探臂，移動到所要的軌道上再試。

BDED-BE03　RDRIGHT

檢驗是否爲正確的軌道。若是，則跳到RTTRK (BE10) 。

若不是，則叫用 SETTRK 副常式 (BE95) 設定正確軌道並將"再尋找"次數予以遞減。

若不爲 0 ，則再尋找軌道。若爲 0 ，則重新調整。

BE04-BE0A　DRVERR

將堆疊和狀態暫存器予以清除。

將 $40 (推動器錯誤) 存入A暫存器。

進入HNDLERR (BEAF) 。

BE0D-BE0F　FORMDSK

跳到DSKFORM (BEAF) 。

BE10-BE25　RTTRK

檢查找到的軌道號數與所要的是否相符。

若冊號未經指定，則不會有錯誤。

若不相符，則將 $20 (冊號不等) 存入 A 暫存器中並經由HNDLERR (BE48) 而跳出。

BE26-BE45　CRCTVOL

檢查段落是否正確。

叫用 ILEAV (BFB8) 來處理段落的軟體交織。

若段落不對，則跳回 TRYADR (BDC1) 再試。

若段落正確，則進一步找出所要做的運算。

若爲寫入，則跳到WRIT(BE51)。
否則，叫用READ16(B8DC)讀取資料。
若讀取良好，則叫用POSTNB16 (B8C2)將資料分塊拼組後，正常地跳回主叫方。

BE46-BE47 ALLDONE
跳過HNDLERR中的"設定進位旗號"指令往下做

BE48-BE50 HNDLERR
設定進位旗號。
將A暫存器存入IOB 中當作返回碼。
將馬達關閉。
跳回主叫方。

BE51-BE59 WRITE
叫用WRITE16 (B8 2A)以寫入一個段落。
若寫入良好，則經由ALLDONE(BE46)而跳出。
若寫入不良，則將$10存入A暫存器(寫入保護)並經由HNDLERR(BE48)而跳出。

BE5A-BE8D MYSEEK
在進入SEEKABS 常式之前，先完成必要的管理工作(Housekeeping)。
決定每軌道的相位數，並將軌道情報存入隨槽路而定的適當位置中。

BE8E-BE94 XTOY常式。
將X暫存器除以16後得出槽路號數放入Y暫存器中

BE95-BEAE 設定軌道號數。

BEAF-BF0C INIT命令處理常式。

為重新設定磁碟做好準備。
從 IOB 取得所要的册號。
將第一及第二緩衝器清為0。
將磁碟探臂重新調整到軌道0。
將段落之間須寫入的同步拜個數設定為$28(40)。
叫用TRACK WRITE 常式來做實際的定格式工作。
在此重新設定的過程中，可允許再試48次。
在叫用TRACK WRITE 之後，重覆檢查所找到的第一個段落為0。
在成功地定好一軌道的格式之後，將軌道號數遞加。
反覆執行，將35個軌道全部定好格式。

BF0D-BF61 TRACK WRITE 常式。
從段落0開始。
一開始是128個自同步拜。
接著是按段落0到段落15的順序。
設定軌道的再查證次數為48。
用正數填入段落的重新設定對映拜。
延遲一段時間，以跳過開頭的自同步拜。
讀取所找到的第一個位址欄。
若讀取良好，且找到段落0，則進入查證軌道(VERIFY TRACK)常式。
將同步拜的計數減2(到16以下為減1)。
若同步拜的個數大於或等於5，則經由BF71跳出。

BF62-BF87 VERIFY TRACK(查證軌道)常式。
此常式從剛定好格式的軌道中讀取所有的16個段落。

若在讀取位址欄或資料欄時有錯誤發生，則將再試次數減1。
若結果不為0，則將繼續讀取。
叫用SECTOR MAP常式(BF88)。

BF88-BFA7 SECTOR MAP(段落對映)常式。
每一段落經過核驗之後，就利用此常式在段落重新設定對映拜上予以標示。
若有錯誤產生，則叫用$BF6C將再試次數減1，、而在其值非0時繼續試。
當軌道0完畢時，將同步拜的計數遞減(若計數大於或等於16時為減2)。

BFA8-BFB7 段落的初始設定對映，用來在初始設定時予以標示。
在軌道初值設定之前，先存入$30。
當每一段落均設定完成後，其值改變為$FF。

BFB8-BFC7 段落轉換表。
利用軟體使段落交織存放。

BFC8-BFD8 程式修補區由此開始。
在開啓過程中，被$B741叫用，以修補零號語言卡。
叫用SETVID($FE93)。
將語言卡的保護解除(若原來有)。
將$00存入$E000。
叫用SETKBD (FE89)並冷起動DOS後跳出

BFD9-BFDB 未使用。

BFDC-BFE5 $A0E2 所叫用的程式修補。
設定三個追加的自然值(拜偏移量=0)。

	跳回主叫方。
BFE6-BFEC	$A6D5所叫用的程式修補。
	叫用$A75B 將狀態重置並設定暖起動旗號。
	將RUN 標示爲不可中斷。
	跳回主叫方。
BFED-BFFF	$B377 所叫用的程式修補。
	叫用$AE7E，將檔案處理程式的工作區予以保存起來。
	存回堆疊中的原值。
	關閉所有開啓的檔（$A316）。
	再將堆疊中的值存入。
	經由$B385（"磁碟客滿錯誤"）而跳出。

DOS第零頁使用情形

拜	用　途
24	水平游標（DOS）
26,27	段落緩衝器位址（ROM）
	臨時使用的空間（RWTS）
28,29	BASL/BASH(DOS)
2A	分段合併計數器（ROM；BOOT）
	臨時使用的空間（RWTS）
2B	BOOT的槽路號數乘以16（ROM）
	臨時使用的空間（RWTS）
2C	段落標頭中的檢核數（RWTS）

2D	段落標頭中的段落號數（RWTS）
2E	段落標頭中的軌道號數（RWTS）
2F	段落標頭中的冊號（RWTS）
33	催促符號（DOS）
35	高比次中存有推動器號數（RWTS）
36,37	CSWL,CSWH（DOS）
38,39	KSWL,KSWH（DOS）
3C	工作拜（ROM）
	合併的工作拜（BOOT）
	裝置特性表位址（RWTS）
3D	段落號數（ROM）
	裝置特性表位址（RWTS）
3E,3F	ROM 中讀取段落副常式的位址（BOOT）
	緩衝器位址（RWTS）
40,41	DOS 對映的位址（BOOT）
	檔案緩衝器位址（DOS）
41	定軌道格式的計數器（RWTS）
42,43	緩衝器位址（DOS）
44,45	數值運算元（DOS）
46,47	臨時使用空間（RWTS）
48,49	IOB 的位址（RWTS）
4A,4B	INTEGER BASIC LOMEN的位址（DOS）
	定磁碟格式的工作空間（RWTS）
4C,4D	INTEGER BASIC HIMEN的位址（DOS）
67,68	APPLESOFT BASIC 程式開始（DOS）
69,6A	APPLESOFT BASIC 變數開始（DOS）

6F,70	APPLESOFT BASIC字串開始(DOS)
73,74	APPLESOFT BASIC HIMEN位址(DOS)
76	APPLESOFT BASIC行數的高位(DOS)
AF,B0	APPLESOFT BASIC程式結束(DOS)
CA,CB	INTEGER BASIC程式開始(DOS)
CC,CD	INTEGER BASIC變數結束(DOS)
D6	APPLESOFT BASIC程式保護旗號(DOS)
D8,D9	INTEGER BASIC行數(DOS)
	APPLESOFT BASIC ONERR(DOS)

附錄A 範例程式

在這一部分裏，將介紹幾個共用程式，可供讀者檢查及修理磁碟。這些程式可作爲如何與DOS內部程式介面的實例。對組合語言不熟悉的讀者可直接將這程式的機器碼經由監督程式輸入磁碟，以方便日後的運用。此處假設採用16個段落的磁碟，但其中大多可很容易修改以配合任何版本的DOS之用。特別建議讀者，除非完全瞭解這些程式，否則應盡量使用備用的或可捨棄的磁碟。因爲不當的叫用程式，雖然不致於造成實體上的損壞，但其中的資料却可能受到破壞，或者須重新設定，相當麻煩。

這裏介紹五個程式：

DUMP 傾印軌道內容。

這是經由輸出／輸入的選擇位址直接觸取磁碟的例子。可用來傾印任一軌道中的原始形式資料到記憶體中，以供檢視，對增進磁碟格式的瞭解和問題的診斷，相當有用。

ZAP　更改磁碟內容。

此程式是修補磁碟目錄表的重要工具。對檢查磁碟上的檔案結構和直接修補檔案或甚至DOS，也極爲有用。此程式亦可供讀取和選擇性地寫入資料到任何段落。因此，也是一個叫用RWTS的好範例。

INIT　重定某單一軌道的格式。

此程式可重新設定磁碟上的某一軌道。任何册號（$00－$FF）均可指定。對任何軌道的段落受到損壞之後，均可用INIT予以重新存回，而不須對整個磁碟重新設定。此程式假設使用DOS3.3及48K記憶體。

FTS　尋找軌道/段落表。

當目錄表受到損壞時，可用FTS補救。FTS尋找磁碟上的每一個段落，發現是軌道/段落表列時，則印出其軌道及段落的位置。一旦知道這些位置，就可利用ZAP予以修補成爲新的目錄表。

COPY　轉錄檔案。

COPY是利用組合語言直接叫用DOS 檔案處理程式的一個例子。可讀取一個B型檔案，將其內容抄錄到一個T型的檔案輸出。雖然，COPY可用來轉換PIE 編輯

程式所使用的檔案以供APPLE TOOLKIT 編輯程式之用，但其本身並不屬於公用程式，而應算是觸取檔案管理式的一個程式。

將程式存入磁碟

有心的讀者可能希望用原始碼寫程式，經由編譯程式將產生的目標碼放入磁碟。APPLE TOOLKIT 編譯程式即可產生此處例舉的報表，有興趣的讀者可參考有關編譯程式在虛擬(Pseudo)運算碼方面的說明。對不使用組合語言的讀者，則可使用下列程序將機器碼輸入。

組合語言的程式報表包含下面幾種情報：

部分目標碼的位址。

應放入該位址的目標碼。

敍述指令號數。

敍述指令本身。

例如：

0800：20DC 03 112 COPY JSR LOCFPL FIND PARMLIST 指出二進位碼"20DC03"應存放在0800，其敍述指令號碼是112。在監督程式中鍵入程式，須同時鍵入其位址及目標碼。譬如把DUMP程式輸入：

CALL -151（從BASIC進入監督程式）

0800：20 E3 03

0803：84 00

0805：85 01

0807：A5 02

等等。

0879：85 35

087B：4C B3 FD

BSAVE：DUMP，A$800，L$7E（將程式存入磁碟）

須注意若某一行（如DUMP中的第四行）沒有目標碼，則可略去。當程式要執行時：

BLOAD DUMP（輸入程式）

CALL -151（進入監督程式）

02：11 N 800G（將軌道存入以備傾印，執行命令）

必須和其他程式一道使用的BSAVE命令有：

BSAVE ZAP，A$900，L$6C

BSAVE INT，A$300，L$89

BSAVE FTS，A$900，L$DC

BSAVE COPY，A$800，L$1EC

包含有這五個程式的磁碟可以用相當合理的價格買到。請直接和下面的地址連絡：

QUALITY SOFTWARE

6660，RESEDA Blvd.，Reseda，Ca.，U.S.A

TEL：(213)344-6599

DUMP－傾印軌道內容

DUMP可將磁碟上任何軌道上的資料以其原始的形式傾印出來，以供檢查段落位址，資料欄和格式的方式。好奇的讀者可因此而對前面幾章所講的觀念更加瞭解清楚。DUMP 也可檢視大多數有保護的磁碟，我們可以以之和通常的比較其差異。對某些段落位址受損或某些輸出／輸入錯誤回復後的資料欄，亦

可用DUMP加以檢視。DUMP是利用組合語言直接控制二號磁碟機(DISKⅡ)的硬體，而很少或根本不用到DOS的一個範例程式。

使用DUMP，首先將欲傾印的軌道號數存在 $02 位置，然後從$800開始執行。在顯示幕上以十六進位顯示出軌道的第一部分之後，跳回監督程式。而整個軌道的映像已存入從$1000開始的位址裏。例如：

CALL -151　(從BASIC進入監督程式)

BLOAD DUMP　(載入DUMP程式)

……將欲傾印的磁碟插入……

02：11 N 800G　(將11(軌道17)存入$02，N結束此存入命令，進入$800)

輸出的結果形式如下：

1000-D5 AA 96 AA AB AA BB AB(段落位址開始)

1008-AA AB BA DE AA E8 CO FF

1010-9E FF FF FF FF FF D5 AA(段落資料開始)

1018-AD AE B2 9D AC AE 96 96(段落資料)

等等。

通常，發生輸出/輸入錯誤現象的段落可能其中只有一個比次不對，可能發生在位址、資料標頭或資料本身。利用DUMP程式再加上時間和耐性，可找出錯誤所在，而再利用ZAP程式予以更正。做這項工作，須先完全了解第三章才可。

```
0800:            2          ORG   $800

0800:            4 ******************************************************
0800:            5 *                                                    *
0800:            6 * DUMP:THIS PROGRAM WILL ALLOW ITS USER TO DUMP AN ENTIRE *
0800:            7 *       TRACK IN ITS RAW FORM INTO MEMORY FOR EXAMINATION. *
0800:            8 *                                                    *
0800:            9 * INPUT: $02 = TRACK TO BE READ                      *
0800:           10 *                                                    *
0800:           11 * OUTPUT:$1000 = ADDRESS OF TRACK IMAGE              *
0800:           12 *                                                    *
0800:           13 * ENTRY POINT: $800                                  *
0800:           14 *                                                    *
0800:           15 * PROGRAMMER: DON D WORTH 2/19/81                    *
0800:           16 *                                                    *
0800:           17 ******************************************************

0800:           19 *        ZPAGE DEFINITIONS

0000:           21 PTR      EQU   $0        WORK POINTER
0002:           22 TRACK    EQU   $2        TRACK TO BE READ/WRITTEN
003C:           23 A1L      EQU   $3C       MONITOR POINTER
003E:           24 A2L      EQU   $3E       MONITOR POINTER
0048:           25 PREG     EQU   $48       MONITOR STATUS REGISTER

0800:           27 *        OTHER ADDRESSES

1000:           29 BUFFER   EQU   $1000     TRACK IMAGE AREA
03E3:           30 LOCRPL   EQU   $3E3      LOCATE RWTS PARMLIST SUBRTN
03D9:           31 RWTS     EQU   $3D9      RWTS SUBROUTINE
FDED:           32 COUT     EQU   $FDED     PRINT ONE CHAR SUBROUTINE
FDB3:           33 XAM      EQU   $FDB3     MONITOR HEX DUMP SUBRTN

0800:           35 *        DISK I/O SELECTS

C080:           37 DRVSM0   EQU   $C080     STEP MOTOR POSITIONS
C081:           38 DRVSM1   EQU   $C081
C082:           39 DRVSM2   EQU   $C082
C083:           40 DRVSM3   EQU   $C083
C084:           41 DRVSM4   EQU   $C084
C085:           42 DRVSM5   EQU   $C085
C086:           43 DRVSM6   EQU   $C086
C087:           44 DRVSM7   EQU   $C087
C088:           45 DRVOFF   EQU   $C088     TURN DRIVE OFF AFTER 6 REVS
C089:           46 DRVON    EQU   $C089     TURN DRIVE ON
C08A:           47 DRVSL1   EQU   $C08A     SELECT DRIVE 1
C08B:           48 DRVSL2   EQU   $C08B     SELECT DRIVE 2
C08C:           49 DRVRD    EQU   $C08C     READ DATA LATCH
C08D:           50 DRVWR    EQU   $C08D     WRITE DATA LATCH
C08E:           51 DRVRDM   EQU   $C08E     SET READ MODE
C08F:           52 DRVWRM   EQU   $C08F     SET WRITE MODE

0800:           54 *        RWTS PARMLIST DEFINITION

0000:           56          DSECT
0000:           57 RPLIOB   DS    1         IOB TYPE ($01)
0001:           58 RPLSLT   DS    1         SLOT*16
0002:           59 RPLDRV   DS    1         DRIVE
0003:           60 RPLVOL   DS    1         VOLUME
0004:           61 RPLTRK   DS    1         TRACK
0005:           62 RPLSEC   DS    1         SECTOR
0006:           63 RPLDCT   DS    2         ADDRESS OF DCT
0008:           64 RPLBUF   DS    2         ADDRESS OF BUFFER
000A:           65 RPLSIZ   DS    2         SECTOR SIZE
000C:           66 RPLCMD   DS    1         COMMAND CODE
0000:           67 RPLCNL   EQU   $00          NULL COMMAND
0001:           68 RPLCRD   EQU   $01          READ COMMAND
0002:           69 RPLCWR   EQU   $02          WRITE COMMAND
0004:           70 RPLCFM   EQU   $04          FORMAT COMMAND
000D:           71 RPLRCD   DS    1         RETURN CODE
```

```
0010:            72 RPLRWP   EQU  $10          WRITE PROTECTED
0020:            73 RPLRVM   EQU  $20          VOLUME MISMATCH
0040:            74 RPLRDE   EQU  $40          DRIVE ERROR
0080:            75 RPLRRE   EQU  $80          READ ERROR
000E:            76 RPLTVL   DS   1         TRUE VOLUME
000F:            77 RPLPSL   DS   1         PREVIOUS SLOT
0010:            78 RPLPDR   DS   1         PREVIOUS DRIVE
C800:            79          DEND

0800:            81 *        USE RWTS TO POSITION THE ARM TO THE DESIRED TRACK

0800:20 E3 03    83 DUMP     JSR  LOCRPL    LOCATE RWTS PARMLIST
0803:84 00       84          STY  PTR       AND SAVE POINTER
0805:85 01       85          STA  PTR+1

0807:A5 02       87          LDA  TRACK     GET TRACK TO READ/WRITE
0809:A0 04       88          LDY  #RPLTRK   STORE IN RWTS LIST
080B:91 00       89          STA  (PTR),Y

080D:A9 00       91          LDA  #RPLCNL   NULL OPERATION
080F:A0 0C       92          LDY  #RPLCMD   AND STORE IN LIST
0811:91 00       93          STA  (PTR),Y

0813:A9 00       95          LDA  #0        ANY VOLUME WILL DO
0815:A0 03                   LDY  #RPLVOL
0817:91 00       97          STA  (PTR),Y
0819:20 E3 03    98          JSR  LOCRPL    RELOAD POINTER TO PARMS
081C:20 D9 03    99          JSR  RWTS      CALL RWTS
081F:A9 00      100          LDA  #0
0821:85 48      101          STA  PREG      FIX P REG SO DOS IS HAPPY

0823:           103 *        PREPARE TO DUMP TRACK TO MEMORY

0823:A0 01      105          LDY  #RPLSLT   GET SLOT*16
0825:B1 00      106          LDA  (PTR),Y
0827:AA         107          TAX
0828:BD 89 C0   108          LDA  DRVON,X   KEEP DRIVE ON
082B:BD 8E C0   109          LDA  DRVRDM,X  INSURE READ MODE

082E:A9 00      111          LDA  #>BUFFER  POINT AT DATA
0830:85 00      112          STA  PTR
0832:A9 10      113          LDA  #<BUFFER
0834:85 01      114          STA  PTR+1
0836:A0 00      115          LDY  #0

0838:           117 *        START DUMPING AT THE BEGINNING OF A SECTOR ADDRESS
0838:           118 *        FIELD OR A SECTOR DATA FIELD

0838:BD 8C C0   120 LOOP1    LDA  DRVRD,X.  WAIT FOR NEXT BYTE
083B:10 FB      121          BPL  LOOP1
083D:C9 FF      122          CMP  #$FF      AUTOSYNC?
083F:D0 F7      123          BNE  LOOP1     NO, DON'T START IN MIDDLE
0841:BD 8C C0   124 LOOP2    LDA  DRVRD,X   WAIT FOR NEXT BYTE
0844:10 FB      125          BPL  LOOP2
0846:C9 FF      126          CMP  #$FF      TWO AUTOSYNCS?
0848:D0 EE      127          BNE  LOOP1     NOT YET
084A:BD 8C C0   128 LOOP3    LDA  DRVRD,X
084D:10 FB      129          BPL  LOOP3
084F:C9 FF      130          CMP  #$FF      STILL AUTOSYNCS?
0851:F0 F7      131          BEQ  LOOP3     YES, WAIT FOR DATA BYTE
0853:D0 05      132          BNE  LOOP4     ELSE, START STORING DATA

0855:           134 *        ONCE ALIGNED, BEGIN COPYING THE TRACK TO MEMORY.
0855:           135 *        COPY AT LEAST TWICE ITS LENGTH TO INSURE WE GET IT
0855:           136 *        ALL.
```

```
0855:BD 8C C0   138 LOOPD    LDA   DRVRD,X    WAIT FOR NEXT DATA BYTE
0858:10 FB      139          BPL   LOOPD
085A:91 00      140 LOOP4    STA   (PTR),Y    STORE IN MEMORY
085C:E6 00      141          INC   PTR        BUMP POINTER
085E:D0 F5      142          BNE   LOOPD
0860:E6 01      143          INC   PTR+1
0862:A5 01      144          LDA   PTR+1
0864:C9 40      145          CMP   #$40       DONE AT LEAST A TRACK?
0866:90 ED      146          BCC   LOOPD      NO, CONTINUE
0868:BD 88 C0   147          LDA   DRVOFF,X   TURN DRIVE OFF

086B:           149 *        WHEN FINISHED, DUMP SOME OF TRACK IN HEX ON SCREEN

086B:A9 00      151 EXIT     LDA   #>BUFFER   DUMP 800.8AF
086D:85 3C      152          STA   A1L
086F:A9 10      153          LDA   #<BUFFER
0871:85 3D      154          STA   A1L+1
0873:A9 AF      155          LDA   #>BUFFER+$AF
0875:85 3E      156          STA   A2L
0877:A9 10      157          LDA   #<BUFFER+$AF
0879:85 3F      158          STA   A2L+1
087B:4C B3 FD   159          JMP   XAM        EXIT VIA HEX DISPLAY

*** SUCCESSFUL ASSEMBLY: NO ERRORS
```

ZAP－更改磁碟內容

由DUMP之後再進一步就是要觸取到磁碟段落層次的資料。ZAP可指定軌道及段落，將其讀入記憶體中，以供檢視及改變，之後再將此段落寫回磁碟。對修補受破壞的檔案目錄表尤其有用。其使用方法在解釋FTS時將詳細說明。

使用ZAP時，須先將所要觸取的軌道和段落號數分別存入$02，$03之中。軌道範圍是$00到$22，而段落範圍則是$00到$0F。例如，欲檢視磁碟的內容表（VTOC）可指定軌道$11及段落$00。在$04中若設定$01，則表示此段落要讀到記憶體中；若設定$02則表示要將記憶體內容寫到磁碟的段落上去，而其他的值則會產生破壞性的結果（例如$04 存入入$04會INIT掉你的磁碟）。當此三個位址設定好之後，即可進入$900執行，ZAP將會從位址$800開開始的256個拜讀進或寫入一個段落。例如：

CALL -151 （從BASIC進入監督程式）

BLOAD ZAP（載入ZAP程式）

……插入要做ZAP的磁碟……

02：11 00 01 N 900G （在$02中存入11（軌道17，目錄軌道），在$03中存入00（段落0），在$04中存入01（讀取），N結束存入命令，900G開始執行ZAP）。

輸出的結果形如：

```
0800-04 11 OF 03 00 00 01 00（VTOC開始）
0808-00 00 00 00 00 00 00 00
0810-00 00 00 00 00 00 00 00
0818-00 00 00 00 00 00 00 00
```

……等等……

在上例中，若偏移量爲3的拜（對此磁碟做INIT的DOS 版本）須要改變，則可如下鍵入：

803：02　　　　　（將03改成02）

04　：02　N　900G（將ZAP改爲寫入模式並執行之）

ZAP會記住原來在$02，$03，$04中的值。

ZAP在讀取段落時所產生的錯誤（也許是輸出／ 輸入錯誤），將會印出其訊息如：

RC ＝ 10

此處的返回碼10，指出磁碟有寫入保護而想做寫入動作。其他的錯誤碼有20－冊號不符，40－推動器錯誤，80－讀取錯誤。參考第六章關於RWTS部分，對這些錯誤有更多的說明。

```
0900:              2          ORG  $900

0900:              4 ***********************************************************
0900:              5 *                                                         *
0900:              6 * ZAP: THIS PROGRAM WILL ALLOW ITS USER TO READ/WRITE     *
0900:              7 *      INDIVIDUAL SECTORS FROM/TO THE DISKETTE            *
0900:              8 *                                                         *
0900:              9 * INPUT: $02 = TRACK TO BE READ                           *
0900:             10 *        $03 = SECTOR TO BE READ/WRITTEN                  *
0900:             11 *        $04 = $01 - READ SECTOR                          *
0900:             12 *              $02 - WRITE SECTOR                         *
0900:             13 *       $800 = ADDRESS OF SECTOR DATA BUFFER              *
0900:             14 *                                                         *
0900:             15 * ENTRY POINT: $900                                       *
0900:             16 *                                                         *
0900:             17 * PROGRAMMER: DON D WORTH 2/15/81                         *
0900:             18 *                                                         *
0900:             19 ***********************************************************

0087:             21 BELL     EQU  $87         BELL CHARACTER

0900:             23 *        ZPAGE DEFINITIONS

0000:             25 PTR      EQU  $0          WORK POINTER
0002:             26 TRACK    EQU  $2          TRACK TO BE READ/WRITTEN
0003:             27 SECTOR   EQU  $3          SECTOR TO BE READ/WRITTEN
0004:             28 OPER     EQU  $4          OPERATION TO BE PERFORMED
0001:             29 READ     EQU  1              READ OPERATION
0002:             30 WRITE    EQU  2              WRITE OPERATION
003C:             31 A1L      EQU  $3C         MONITOR POINTER
003E:             32 A2L      EQU  $3E         MONITOR POINTER
0048:             33 PREG     EQU  $48         MONITOR STATUS REGISTER

0900:             35 *        OTHER ADDRESSES
```

```
0800:          37 BUFFER   EQU  $800      SECTOR DATA BUFFER
03E3:          38 LOCRPL   EQU  $3E3      LOCATE RWTS PARMLIST SUBRTN
03D9:          39 RWTS     EQU  $3D9      RWTS SUBROUTINE
FDED:          40 COUT     EQU  $FDED     PRINT ONE CHAR SUBROUTINE
FDDA:          41 PRBYTE   EQU  $FDDA     PRINT ONE HEX BYTE SUBRTN
FDB3:          42 XAM      EQU  $FDB3     MONITOR HEX DUMP SUBRTN

0900:          44 *        RWTS PARMLIST DEFINITION

0000:          46          DSECT
0000:          47 RPLIOB   DS   1         IOB TYPE ($01)
0001:          48 RPLSLT   DS   1         SLOT*16
0002:          49 RPLDRV   DS   1         DRIVE
0003:          50 RPLVOL   DS   1         VOLUME
0004:          51 RPLTRK   DS   1         TRACK
0005:          52 RPLSEC   DS   1         SECTOR
0006:          53 RPLDCT   DS   2         ADDRESS OF DCT.
0008:          54 RPLBUF   DS   2         ADDRESS OF BUFFER
000A:          55 RPLSIZ   DS   2         SECTOR SIZE
000C:          56 RPLCMD   DS   1         COMMAND CODE
0000:          57 RPLCNL   EQU  $00          NULL COMMAND

0001:          58 RPLCRD   EQU  $01          READ COMMAND
0002:          59 RPLCWR   EQU  $02          WRITE COMMAND
0004:          60 RPLCFM   EQU  $04          FORMAT COMMAND
000D:          61 RPLRCD   DS   1         RETURN CODE
0010:          62 RPLRWP   EQU  $10          WRITE PROTECTED
0020:          63 RPLRVM   EQU  $20          VOLUME MISMATCH
0040:          64 RPLRDE   EQU  $40          DRIVE ERROR
0080:          65 RPLRRE   EQU  $80          READ ERROR
000E:          66 RPLTVL   DS   1         TRUE VOLUME
000F:          67 RPLPSL   DS   1         PREVIOUS SLOT
0010:          68 RPLPDR   DS   1         PREVIOUS DRIVE
0900:          69          DEND


0900:          71 *        FILL IN RWTS LIST

0900:20 E3 03  73 ZAP      JSR  LOCRPL    LOCATE RWTS PARMLIST
0903:84 00     74          STY  PTR       AND SAVE POINTER
0905:85 01     75          STA  PTR+1

0907:A5 02     77          LDA  TRACK     GET TRACK TO READ/WRITE
0909:A0 04     78          LDY  #RPLTRK   STORE IN RWTS LIST
090B:91 00     79          STA  (PTR),Y

090D:A5 03     81          LDA  SECTOR    GET SECTOR TO READ/WRITE
090F:C9 10     82          CMP  #16       BIGGER THAN 16 SECTORS?
0911:90 04     83          BCC  SOK       NO
0913:A9 00     84          LDA  #0
0915:85 03     85          STA  SECTOR    YES, PUT IT BACK TO ZERO
0917:A0 05     86 SOK      LDY  #RPLSEC
0919:91 00     87          STA  (PTR),Y   STORE IN RWTS LIST

091B:A0 08     89          LDY  #RPLBUF
091D:A9 00     90          LDA  #>BUFFER  STORE BUFFER PTR IN LIST
091F:91 00     91          STA  (PTR),Y
0921:C8        92          INY
0922:A9 08     93          LDA  #<BUFFER
0924:91 00     94          STA  (PTR),Y

0926:A5 04     96          LDA  OPER      GET COMMAND CODE
0928:A0 0C     97          LDY  #RPLCMD   AND STORE IN LIST
092A:91 00     98          STA  (PTR),Y

092C:A9 00    100          LDA  #0        ANY VOLUME WILL DO
092E:A0 03    101          LDY  #RPLVOL
0930:91 00    102          STA  (PTR),Y

0932:         104 *        NOW CALL RWTS TO READ/WRITE THE SECTOR
```

```
0932:20 E3 03  106          JSR  LOCRPL     RELOAD POINTER TO PARMS
0935:20 D9 03  107          JSR  RWTS       CALL RWTS
0938:A9 00     108          LDA  #0
093A:85 48     109          STA  PREG       FIX P REG SO DOS IS HAPPY
093C:90 1B     110          BCC  EXIT       ALL IS WELL

093E:          112 *        ERROR OCCURED   PRINT "RC=XX"

093E:A9 87     114          LDA  #BELL      BEEP THE SPEAKER
0940:20 ED FD  115          JSR  COUT
0943:A9 D2     116          LDA  #'R        PRINT THE "RC="
0945:20 ED FD  117          JSR  COUT
0948:A9 C3     118          LDA  #'C
094A:20 ED FD  119          JSR  COUT
094D:A9 BD     120          LDA  #'=
094F:20 ED FD  121          JSR  COUT
0952:A0 0D     122          LDY  #RPLRCD
0954:B1 00     123          LDA  (PTR),Y    GET RWTS RETURN CODE
0956:20 DA FD  124          JSR  PRBYTE     PRINT RETURN CODE IN HEX

0959:          126 *        WHEN FINISHED, DUMP SOME OF SECTOR IN HEX

0959:A9 00     128 EXIT     LDA  #>BUFFER   DUMP 800.8B7
095B:85 3C     129          STA  A1L
095D:A9 08     130          LDA  #<BUFFER
095F:85 3D     131          STA  A1L+1
0961:A9 AF     132          LDA  #>BUFFER+$AF
0963:85 3E     133          STA  A2L
0965:A9 08     134          LDA  #<BUFFER+$AF
0967:85 3F     135          STA  A2L+1
0969:4C B3 FD  136          JMP  XAM        EXIT VIA HEX DISPLAY

*** SUCCESSFUL ASSEMBLY: NO ERRORS
```

INIT－對單一軌道重定格式

有時候，磁碟上的分段情報資料受到損壞，以致DOS無法找到其中某個段落。補救的方法是把受到影響的軌道中的段落位址和資料欄全部重定格式。INIT就是用來選擇性地對單一軌道重定格式，而不須對整個磁碟重新設定。在使用INIT之前，應先在此有問題的軌道上嘗試寫入（利用ZAP程式），若RWTS拒絕寫入（返回碼＝40）才須對此整個軌道執行INIT。爲了避免資料漏失，軌道上其他所有的段落最好在INIT之前，抄錄到另一磁碟上，而在完成之後再抄錄回此修補過的磁碟上，則資料就可寫入到原先受破壞的段落中去。

執行INIT之前，須先將所要重定格式的軌道號數存入$02，册號存入$03（册號應與其他軌道的册號相符），而從$800開始執行。INIT在執行完畢之後，會跳回監督程式。若因某些理由（如：磁碟硬體的損壞或推動器本身的問題）而無法重新定格式，則將會印出返回碼以指出錯誤。例如：

CALL -151　　（從BASIC進入監督程式）

BLOAD INIT　（將INIT程式載入）

……插入要INIT的磁碟……

02：11 FE N 800G（在$02中存入11（軌道17，目錄軌道），$03中存入册號$FE（254），N結束存入命令，進入$800開始執行）。

警告：在執行INIT之前，DOS3.3必須載入且須爲48K APPLE。對其他版本的DOS或其他的記憶容量大小，INIT均不會作用。

```
0800:           2          ORG  $800

0800:           4 ************************************************************
0800:           5 *                                                          *
0800:           6 * INIT: THIS PROGRAM WILL ALLOW ITS USER TO INITIALIZE A   *
0800:           7 *       SINGLE TRACK WITH ANY VOLUME NUMBER DESIRED.       *
0800:           8 *                                                          *
0800:           9 * INPUT: $02 = TRACK TO BE INITIALISED                     *
0800:          10 *                                                          *
0800:          11 *        $03 = VOLUME NUMBER                               *
0800:          12 *                                                          *
0800:          13 * ENTRY POINT: $800                                        *
0800:          14 *                                                          *
0800:          15 * PROGRAMMER: PIETER LECHNER 2/19/81                       *
0800:          16 *                                                          *
0800:          17 ************************************************************

0800:          19 *        ZPAGE DEFINITIONS

0000:          21 PTR      EQU  $0        WORK POINTER
0002:          22 TRACK    EQU  $2        TRACK TO BE READ/WRITTEN
0003:          23 VOLUME   EQU  $3        VOLUME NUMBER
002D:          24 SECFND   EQU  $2D       SECTOR FOUND BY RDADR16
003E:          25 AA       EQU  $3E       ZPAGE CONSTANT FOR TIMING
0041:          26 VOL      EQU  $41       VOLUME USED BY WRADR16
0044:          27 TRK      EQU  $44       TRACK USED BY WRADR16
0045:          28 SYNCNT   EQU  $45       SYNC COUNT USED BY DSKF2
0048:          29 PREG     EQU  $48       MONITOR P REGISTER SAVEAREA
0087:          30 BELL     EQU  $87       ASCII BELL

0800:          32 *        OTHER ADDRESSES

03E3:          34 LOCRPL   EQU  $3E3      LOCATE RWTS PARMLIST SUBRTN
03D9:          35 RWTS     EQU  $3D9      RWTS SUBROUTINE
0578:          36 RTRYCNT  EQU  $578      RETRY COUNT FOR DSKF2
BB00:          37 NBUF1    EQU  $BB00     PRIMARY SECTOR BUFFER
BC00:          38 NBUF2    EQU  $BC00     SECONDARY SECTOR BUFFER
B8DC:          39 READ16   EQU  $B8DC     READ DATA FIELD ROUTINE
B944:          40 RDADR16  EQU  $B944     READ ADDRESS FIELD ROUTINE
BF0D:          41 DSKF2    EQU  $BF0D     FORMAT ONE TRACK ROUTINE
FDED:          42 COUT     EQU  $FDED     MONITOR CHARACTER OUTPUT
FDDA:          43 PRBYTE   EQU  $FDDA     MONITOR HEX OUTPUT

0800:          45 *        DISK I/O SELECTS

C080:          47 DRVSM0   EQU  $C080     STEP MOTOR POSITIONS
C081:          48 DRVSM1   EQU  $C081
C082:          49 DRVSM2   EQU  $C082
C083:          50 DRVSM3   EQU  $C083
C084:          51 DRVSM4   EQU  $C084
C085:          52 DRVSM5   EQU  $C085
C086:          53 DRVSM6   EQU  $C086
C087:          54 DRVSM7   EQU  $C087
C088:          55 DRVOFF   EQU  $C088     TURN DRIVE OFF AFTER 6 REVS
C089:          56 DRVON    EQU  $C089     TURN DRIVE ON
C08A:          57 DRVSL1   EQU  $C08A     SELECT DRIVE 1
C08B:          58 DRVSL2   EQU  $C08B     SELECT DRIVE 2
C08C:          59 DRVRD    EQU  $C08C     READ DATA LATCH
C08D:          60 DRVWR    EQU  $C08D     WRITE DATA LATCH
C08E:          61 DRVRDM   EQU  $C08E     SET READ MODE
C08F:          62 DRVWRM   EQU  $C08F     SET WRITE MODE

0800:          64 *        RWTS PARMLIST DEFINITION
```

```
0000:              66          DSECT
0000:              67 RPLIOB   DS    1          IOB TYPE ($01)
0001:              68 RPLSLT   DS    1          SLOT*16
0002:              69 RPLDRV   DS    1          DRIVE
0003:              70 RPLVOL   DS    1          VOLUME
0004:              71 RPLTRK   DS    1          TRACK
0005:              72 RPLSEC   DS    1          SECTOR
0006:              73 RPLDCT   DS    2          ADDRESS OF DCT
0008:              74 RPLBUF   DS    2          ADDRESS OF BUFFER
000A:              75 RPLSIZ   DS    2          SECTOR SIZE

000C:              76 RPLCMD   DS    1          COMMAND CODE
0000:              77 RPLCNL   EQU   $00           NULL COMMAND
0001:              78 RPLCRD   EQU   $01           READ COMMAND
0002:              79 RPLCWR   EQU   $02           WRITE COMMAND
0004:              80 RPLCFM   EQU   $04           FORMAT COMMAND
000D:              81 RPLRCD   DS    1          RETURN CODE
0010:              82 RPLRWP   EQU   $10           WRITE PROTECTED
0020:              83 RPLRVM   EQU   $20           VOLUME MISMATCH
0040:              84 RPLRDE   EQU   $40           DRIVE ERROR
0080:              85 RPLRRE   EQU   $80           READ ERROR
000E:              86 RPLTVL   DS    1          TRUE VOLUME
000F:              87 RPLPSL   DS    1          PREVIOUS SLOT
0010:              88 RPLPDR   DS    1          PREVIOUS DRIVE
0800:              89          DEND

0800:              91 *        USE RWTS TO POSITION THE ARM TO THE DESIRED TRACK

0800:20 E3 03      93 DUMP     JSR   LOCRPL     LOCATE RWTS PARMLIST
0803:84 00         94          STY   PTR        AND SAVE POINTER
0805:85 01         95          STA   PTR+1

0807:A5 02         97          LDA   TRACK      GET TRACK TO READ/WRITE
0809:A0 04         98          LDY   #RPLTRK    STORE IN RWTS LIST
080B:91 00         99          STA   (PTR),Y

080D:A9 00        101          LDA   #RPLCNL    NULL OPERATION
080F:A0 0C        102          LDY   #RPLCMD    AND STORE IN LIST
0811:91 00        103          STA   (PTR),Y

0813:A9 00        105          LDA   #0         ANY VOLUME WILL DO
0815:A0 03        106          LDY   #RPLVOL
0817:91 00        107          STA   (PTR),Y
0819:20 E3 03     108          JSR   LOCRPL     RELOAD POINTER TO PARMS
081C:20 D9 03     109          JSR   RWTS       CALL RWTS
081F:BD 89 C0     110          LDA   DRVON,X    LEAVE DRIVE ON

0822:             112 *        ESTABLISH ENVIRONMENT FOR DSKF2 ROUTINE

0822:A5 02        114          LDA   TRACK      PASS TRACK TO DSKF2
0824:85 44        115          STA   TRK

0826:A5 03        116          LDA   VOLUME     AND VOLUME
0828:85 41        117          STA   VOL
082A:A9 AA        118          LDA   #$AA       STORE CONSTANT FOR ZPAGE..
082C:85 3E        119          STA   AA         TIMING
082E:A9 28        120          LDA   #$28       START WITH 40 SYNCS..
0830:85 45        121          STA   SYNCNT     BETWEEN SECTORS
0832:A0 56        122          LDY   #$56
0834:A9 00        123          LDA   #$00
0836:99 FF BB     124 ZNBUF2   STA   NBUF2-1,Y  ZERO SECONDARY BUFFER
0839:88           125          DEY
083A:D0 FA        126          BNE   ZNBUF2
083C:99 00 BF     127 ZNBUF1   STA   NBUF1,Y    AND PRIMARY BUFFER
083F:88           128          DEY
0840:D0 FA        129          BNE   ZNBUF1

0842:             131 *        INITIALIZE TRACK
```

```
0842:20 0D BF   133          JSR  DSEF2      FORMAT TRACK AND VERIFY
0845:A9 08      134          LDA  #$08       IN CASE OF ERROR...
0847:B0 19      135          BCS  HNDERR     ERROR?

0849:           137 *        READ SECTOR ZERO TO VERIFY FORMATTING

0849:A9 30      139          LDA  #$30       NO, DOUBLE CHECK TRACK
084B:8D 78 05   140          STA  RTRYCNT    ALLOW 48 RETRIES
084E:38         141 NOGOOD   SEC
084F:CE 78 05   142          DEC  RTRYCNT    COUNT RETRIES
0852:F0 0E      143          BEQ  HNDERR
0854:20 44 B9   144          JSR  RDADR16    READ AN ADDRESS FIELD
0857:B0 F5      145          BCS  NOGOOD     ERROR, TRY AGAIN
0859:A5 2D      146          LDA  SECFND     IS THIS SECTOR ZERO?
085B:D0 F1      147          BNE  NOGOOD     NO, TRY AGAIN

085D:20 DC B8   148          JSR  READ16     YES, READ DATA FIELD
0860:90 1F      149          BCC  DONETRK    ALL IS WELL, DONE.
0862:A0 0D      150 HNDERR   LDY  #RPLRCD    ELSE, PHONEY UP A RC
0864:91 0C      151          STA  (PTR),Y

0866:           153 *        ERROR OCCURED, PRINT "RC=XX"

0866:A9 87      155          LDA  #BELL      BEEP THE SPEAKER
0868:20 ED FD   156          JSR  COUT
086B:A9 D2      157          LDA  #'R        PRINT THE "RC="
086D:20 ED FD   158          JSR  COUT
0870:A9 C3      159          LDA  #'C
0872:20 ED FD   160          JSR  COUT
0875:A9 BD      161          LDA  #'=
0877:20 ED FD   162          JSR  COUT
087A:A0 0D      163          LDY  #RPLRCD
087C:B1 00      164          LDA  (PTR),Y    GET RWTS RETURN CODE
087E:20 DA FD   165          JSR  PRBYTE     PRINT RETURN CODE IN HEX

0881:           167 *        WHEN DONE, EXIT TO CALLER

0881:BD 88 C0   169 DONETRK  LDA  DRVOFF,X   TURN DRIVE OFF
0884:A9 00      170          LDA  #$00
0886:85 48      171          STA  PREG       CLEAR P REGISTER FOR DOS
0888:60         172          RTS             ; RETURN TO CALLER
```

FTS－找出T/S表

磁碟在目錄軌道中突然發生輸出／輸入錯誤的機會是經常難免的。萬一不幸，則無論怎樣使用磁碟，都會收到DOS送出的I／O錯誤訊息。通常，在這種情形下，磁碟檔案中所存的資料仍然是完整的，只是其指標找不到而已。若是一定要恢復這些資料，則必須要重建目錄。首先要找出每一檔案的所有T/S表，然後利用ZAP程式對每一個找到的檔案在軌道17中修補一個目錄的進入點。FTS即爲從磁碟中掃瞄找出T／S表的共用程式。雖然，FTS可能會將某些形式相同的段落誤認爲T/S表但絕不會漏掉眞正的T/S表。因此，在做完FTS之後，必須再用ZAP來檢查FTS所印出的每一個軌道／段落，看看是不是眞的T／S表。另外，對某些已經被刪除掉的檔案所用到的T／S表，FTS也會將其找出來。因爲不太容易判斷那一個檔案是有效那一個是被刪除掉的，所以通常有必要將所有的檔案先恢復起來並抄錄到另一磁碟上，而後再將重覆的或不要的予以刪除。

執行FTS，只要將其程式載入，並從$900開始執行。FTS會印出它所找到與T／S表相似的每一個段落的軌道和段落號數。例如：

CALL -151 （從BASIC進入監督程式）

BLOAD FTS （載入FTS程式）

……插入要掃瞄的磁碟……

900G （執行FTS）

輸出的結果形如：

T = 12 S = 0F

T = 13　S = 0F

T = 14　S = 0D

T = 14　S = 0F

這裏，只找到4個可能的檔案。再用 ZAP 讀取軌道 $12，段落$0F，而在此段落上的+$0C 位址中存有此檔案第一個段落的軌道和段落位址。檢視該段落，可以識別檔案及其型式。通常，一個BASIC程式可以從其PRINT 敍述中包含的文字串而加以識別。利用ASCII 轉換表(請參考APPLE Ⅱ REFERENCE MANUAL第8頁)可以將這些字串解碼出來。純粹T型檔案也包含有ASCII文字，其中每一行與另一行之間有一個$8D(回車)加以分隔。B型檔案最不容易辨認，除非在頭4個拜中所存的位址和長度能認得出來。若檔案無法辨認，可先假設爲APPLESOFT BASIC。當後來發覺假設不對，再回頭用 ZAP 處理目錄中的檔案型式，做另外的嘗試。下面是一個例子，說明ZAP如何從一個檔案的T/S表(T=12，S=0F)建立其目錄進入點。

```
CALL -151
BLOAD ZAP
……插入要做ZAP的磁碟……
800:00 N 801<800.8FEM (將記憶體的段落區清爲0)
80B:12 0F 02(軌道12，段落OF，型式A)
:C1 A0 A0 A0 A0 A0 A0 (名字爲“A”)
:A0 A0 A0 A0 A0 A0 A0 (以29個空白將檔名填滿)
:A0 A0 A0 A0 A0 A0 A0
:A0 A0 A0 A0 A0 A0 A0
:A0 A0
```

02：11　0F　02　N　900G　（將此新的段落對映當作第一個（且為僅有的）目錄段落予以寫出）

此檔案應立即抄錄到另一磁碟，然後重覆處理所有其他由FTS找出的T/S表，直到所有檔案恢復為止。當每一個檔案恢復之後，可用RENAME命令改回到原來的名字。等到所有的檔案都抄錄到另一磁碟，並經試測為正常之後，就可將該受破壞的磁碟重新設定（INIT）。

```
0900:           2          ORG  $900

0900:           4 ***********************************************************
0900:           5 *                                                         *
0900:           6 * FTS: THIS PROGRAM SCANS THE ENTIRE DISKETTE FOR WHAT    *
0900:           7 *      APPEAR TO BE TRACK/SECTOR LISTS AND PRINTS THE     *
0900:           8 *      TRACK AND SECTOR OF EACH ONE IT FINDS.             *
0900:           9 *                                                         *
0900:          10 * INPUT: NONE                                             *
0900:          11 *                                                         *
0900:          12 * ENTRY POINT: $900                                       *
0900:          13 *                                                         *
0900:          14 * PROGRAMMER: DON D WORTH 2/15/81                         *
0900:          15 *                                                         *
0900:          16 ***********************************************************

0087:          18 BELL     EQU  $87        BELL CHARACTER
008D:          19 RETURN   EQU  $8D        CARRIAGE RETURN

0900:          21 *        ZPAGE DEFINITIONS

0000:          23 PTR      EQU  $0         WORK POINTER
003C:          24 A1L      EQU  $3C        MONITOR POINTER
003E:          25 A2L      EQU  $3E        MONITOR POINTER
0048:          26 PREG     EQU  $48        MONITOR STATUS REGISTER

0900:          28 *        OTHER ADDRESSES

0800:          30 BUFFER   EQU  $800       SECTOR DATA BUFFER
03E3:          31 LOCRPL   EQU  $3E3       LOCATE RWTS PARMLIST SUBRTN
03D9:          32 RWTS     EQU  $3D9       RWTS SUBROUTINE
FDED:          33 COUT     EQU  $FDED      PRINT ONE CHAR SUBROUTINE
FDDA:          34 PRBYTE   EQU  $FDDA      PRINT ONE HEX BYTE SUBRTN

0900:          36 *        RWTS PARMLIST DEFINITION

0000:          38          DSECT
0000:          39 RPLIOB   DS   1          IOB TYPE ($01)
0001:          40 RPLSLT   DS   1          SLOT*16
0002:          41 RPLDRV   DS   1          DRIVE
0003:          42 RPLVOL   DS   1          VOLUME
0004:          43 RPLTRK   DS   1          TRACK
0005:          44 RPLSEC   DS   1          SECTOR
```

```
0006:             45 RPLDCT  DS   2          ADDRESS OF DCT
0008:             46 RPLBUF  DS   2          ADDRESS OF BUFFER
000A:             47 RPLSIZ  DS   2          SECTOR SIZE
000C:             48 RPLCMD  DS   1          COMMAND CODE
0000:             49 RPLCNL  EQU  $00           NULL COMMAND
0001:             50 RPLCRD  EQU  $01           READ COMMAND
0002:             51 RPLCWR  EQU  $02           WRITE COMMAND
0004:             52 RPLCFM  EQU  $04           FORMAT COMMAND
000D:             53 RPLRCD  DS   1          RETURN CODE
0010:             54 RPLRWP  EQU  $10           WRITE PROTECTED
0020:             55 RPLRVM  EQU  $20           VOLUME MISMATCH
0040:             56 RPLRDE  EQU  $40           DRIVE ERROR
0080:             57 RPLRRE  EQU  $80           READ ERROR
000E:             58 RPLTVL  DS   1          TRUE VOLUME
000F:             59 RPLPSL  DS   1          PREVIOUS SLOT
0010:             60 RPLPDR  DS   1          PREVIOUS DRIVE
0900:             61         DEND

0900:             63 *       START TRACK/SECTOR JUST PAST DOS (TRACK 3)

0900:20 E3 03     65 FTS     JSR  LOCRPL     LOCATE RWTS PARMLIST
0903:84 00        66         STY  PTR        AND SAVE POINTER
0905:85 01        67         STA  PTR+1

0907:A9 03        69         LDA  #3         FIRST NON-DOS TRACK
0909:A0 04        70         LDY  #RPLTRK    STORE IN RWTS LIST
090B:91 00        71         STA  (PTR),Y

090D:A0 08        73         LDY  #RPLBUF
090F:A9 00        74         LDA  #>BUFFER   STORE BUFFER PTR IN LIST
0911:91 00        75         STA  (PTR),Y
0913:C8           76         INY
0914:A9 09        77         LDA  #<BUFFER
0916:91 00        78         STA  (PTR),Y

0918:A9 01        80         LDA  #RPLCRD    GET COMMAND CODE FOR READ
091A:A0 0C        81         LDY  #RPLCMD    AND STORE IN LIST
091C:91 00        82         STA  (PTR),Y

091E:A9 00        84         LDA  #0         ANY VOLUME WILL DO
0920:A0 03        85         LDY  #RPLVOL
0922:91 00        86         STA  (PTR),Y

0924:             88 *       NEW TRACK, START SECTOR AT ZERO

0924:A0 05        90 NEWTRK  LDY  #RPLSEC
0926:A9 00        91         LDA  #0
0928:91 00        92         STA  (PTR),Y

092A:             94 *       NOW CALL RWTS TO READ THE SECTOR

092A:20 E3 03     96 NEWSEC  JSR  LOCRPL     RELOAD POINTER TO PARMS
092D:20 D9 03     97         JSR  RWTS       CALL RWTS
0930:A9 00        98         LDA  #0
0932:85 48        99         STA  PREG       FIX P REG SO DOS IS HAPPY
0934:90 26       100         BCC  SCAN       ALL IS WELL

0936:            102 *       ERROR OCCURED, PRINT "RC=XX"

0936:20 B3 C9    104         JSR  PRTTS      PRINT TRACK/SECTOR
0939:A9 87       105         LDA  #BELL      BEEP THE SPEAKER
093B:20 ED FD    106         JSR  COUT
093E:A9 D2       107         LDA  #'R        PRINT THE "RC="
0940:20 ED FD    108         JSR  COUT
0943:A9 C3       109         LDA  #'C
0945:20 ED FD    110         JSR  COUT
0948:A9 BD       111         LDA  #'=
094A:20 ED FD    112         JSR  COUT
094D:A0 0D       113         LDY  #RPLRCD
094F:B1 00       114         LDA  (PTR),Y    GET RWTS RETURN CODE
```

```
0951:20 DA FD   115          JSR  PRBYTE     PRINT RETURN CODE IN HEX
0954:A9 8D      116          LDA  #RETURN
0956:20 ED FD   117          JSR  COUT
0959:4C 8E 09   118          JMP  NXTSEC     GO ON

095C:           120 *        NO ERROR, SEE IF SECTOR LOOKS LIKE A T/S LIST

095C:A2 00      122 SCAN     LDX  #0
095E:BD 00 08   123 SCLP0    LDA  BUFFER,X   MAKE SURE ITS NOT ALL ZERO
0961:D0 05      124          BNE  SCAN1
0963:E8         125          INX
0964:D0 F8      126          BNE  SCLP0
0966:F0 26      127          BEQ  NXTSEC     IF IT IS, SKIP IT

0968:A2 05      129 SCAN1    LDX  #5         START AT OFFSET 5
096A:BD 00 08   130 SCLP1    LDA  BUFFER,X
096D:D0 1F      131          BNE  NXTSEC     HEADER OF T/S MUST BE ZERO
096F:E8         132          INX
0970:E0 0C      133          CPX  #12        AT THE T/S PAIRS YET?
0972:90 F6      134          BCC  SCLP1      NO, KEEP CHECKING

0974:BD 00 08   136 SCLP2    LDA  BUFFER,X   GET TRK
0977:C9 23      137          CMP  #35        MUST BE 0-34
0979:B0 13      138          BCS  NXTSEC
097B:E8         139          INX
097C:BD 00 08   140          LDA  BUFFER,X   GET SECTOR
097F:C9 10      141          CMP  #16        MUST BE 0-15
0981:B0 0B      142          BCS  NXTSEC
0983:E8         143          INX
0984:D0 EE      144          BNE  SCLP2

0986:20 B3 09   146          JSR  PRTTS      ALL CONDITIONS MET
0989:A9 8D      147          LDA  #RETURN
098B:20 ED FD   148          JSR  COUT

098E:           150 *        BUMP SECTOR NUMBER OR TRACK AND CONTINUE

098E:A0 05      152 NXTSEC   LDY  #RPLSEC
0990:B1 00      153          LDA  (PTR),Y    GET LAST SECTOR
0992:18         154          CLC
0993:69 01      155          ADC  #1         BUMP BY ONE
0995:91 00      156          STA  (PTR),Y    AND PUT IT BACK IN LIST
0997:C9 10      157          CMP  #16        TOO BIG?
0999:B0 03      158          BCS  NXTTRK
099B:4C 2A 09   159          JMP  NEWSEC     NO, GO READ IT

099E:A0 04      161 NXTTRK   LDY  #RPLTRK
09A0:B1 00      162          LDA  (PTR),Y    GET LAST TRACK
09A2:18         163          CLC
09A3:69 01      164          ADC  #1         BUMP BY ONE
09A5:91 00      165          STA  (PTR),Y    AND PUT IT BACK IN LIST
09A7:C9 11      166          CMP  #$11       CATALOG TRACK?
09A9:F0 F3      167          BEQ  NXTTRK     YES, SKIP OVER THAT ONE
09AB:C9 23      168          CMP  #35        DONE ALL 35 TRACKS?
09AD:B0 03      169          BCS  EXIT       YES, LEAVE
09AF:4C 24 09   170          JMP  NEWTRK     NO, GO READ FIRST SECTOR
09B2:60         171 EXIT     RTS

09B3:           173 *        PRTTS: PRINT "T=XX S=XX"

09B3:A9 D4      175 PRTTS    LDA  #'T        PRINT "T"
09B5:20 ED FD   176          JSR  COUT
09B8:A0 04      177          LDY  #RPLTRK
09BA:B1 00      178          LDA  (PTR),Y
09BC:20 CC 09   179          JSR  PRTEQ      PRINT "=XX "

09BF:A9 D3      181          LDA  #'S        PRINT "S"
09C1:20 ED FD   182          JSR  COUT
09C4:A0 05      183          LDY  #RPLSEC
09C6:B1 00      184          LDA  (PTR),Y
09C8:20 CC 09   185          JSR  PRTEQ      PRINT "=XX "
09CB:60         186          RTS
```

```
09CC:48          188 PRTEQ   PHA
09CD:A9 BD       189         LDA  #'=
09CF:20 ED FD    190         JSR  COUT
09D2:68          191         PLA
09D3:20 DA FD    192         JSR  PRBYTE
09D6:A9 AC       193         LDA  #',
09D8:20 ED FD    194         JSR  COUT
09DB:60          195         RTS

*** SUCCESSFUL ASSEMBLY: NO ERRORS
```

COPY—檔案轉錄

此程式可當作示範如何用組合語言來使用DOS 檔案處理程式中副程式的範例。COPY將B型檔案當作輸入讀進，去掉位址和長度情報，將其資料當作新產生的T型檔案予以寫出。輸入檔的名稱設爲“ INPUT ”（雖然此一名字很容易被鍵入），輸出檔名稱設爲“ OUTPUT ”。COPY是單推動器的操作，使用最後叫到的推動器。

執行COPY時，先將其載入，然後從$800開始執行：

CALL -151　　（從BASIC進入監督程式）

BLOAD COPY（載入COPY程式）

……插入包含有INPUT檔的磁碟……

800G　　（執行COPY ）

當COPY完成，會回到BASIC。若有錯誤產生，則經由檔案處理程式送回的返回碼會印出來。可參考第六章關於檔案處理程式的返回碼而知其原因。

```
0800:          2        ORG   $800

0800:          4 ***********************************************************
0800:          5 *                                                         *
0800:          6 * COPY:THIS PROGRAM DEMONSTRATES THE USE OF THE DOS FILE  *
0800:          7 *      MANAGER BY COPYING A BINARY FILE TO A TEXT FILE.   *
0800:          8 *                                                         *
0800:          9 * INPUT: INPUT FILE NAME IS "INPUT"                       *
0800:         10 *        OUTPUT FILE NAME IS "OUTPUT"                     *
0800:         11 *                                                         *
0800:         12 * ENTRY POINT: $800                                       *
0800:         13 *                                                         *
0800:         14 * PROGRAMMER: DON D WORTH 2/19/81                         *
0800:         15 *                                                         *
0800:         16 ***********************************************************

0087:         18 BELL     EQU  .$87       BELL CHARACTER

0800:         20 *        ZPAGE DEFINITIONS

0000:         22 PTR      EQU  $0         WORK POINTER
0002:         23 BUFP     EQU  $2         BUFFER POINTER
0004:         24 EBYTE    EQU  $4
003C:         25 A1L      EQU  $3C        MONITOR POINTER
003E:         26 A2L      EQU  $3E        MONITOR POINTER
```

```
0800:           28 *            OTHER ADDRESSES

1000:           30 BUFFER  EQU  $1000      DATA BUFFER
03D0:           31 DOSWRM  EQU  $3D0       DOS WARMSTART ADDRESS
03E3:           32 LOCRPL  EQU  $3E3       LOCATE RWTS PARMLIST SUBRTN
03DC:           33 LOCFPL  EQU  $3DC       LOCATE FILE MGR PARMLIST SUB
03D6:           34 FM      EQU  $3D6       FILE MANAGER ENTRY POINT
FDED:           35 COUT    EQU  $FDED      PRINT ONE CHAR SUBROUTINE
FDDA:           36 PRBYTE  EQU  $FDDA      PRINT ONE HEX BYTE SUBRTN

0800:           38 *            FILE MANAGER PARMLIST DEFINITION

0000:           40         DSECT
0000:           41 FMOCOD  DS   1          OPERATION CODE
0001:           42 FMOCOP  EQU  $01           OPEN
0002:           43 FMOCCL  EQU  $02           CLOSE
0003:           44 FMOCRD  EQU  $03           READ
0004:           45 FMOCWR  EQU  $04           WRITE
0005:           46 FMOCDE  EQU  $05           DELETE
0006:           47 FMOCCA  EQU  $06           CATALOG
0007:           48 FMOCLO  EQU  $07           LOCK
0008:           49 FMOCUN  EQU  $08           UNLOCK
0009:           50 FMOCRE  EQU  $09           RENAME
000A:           51 FMOCPO  EQU  $0A           POSITION
000B:           52 FMOCIN  EQU  $0B           INIT
000C:           53 FMOCVE  EQU  $0C           VERIFY
0001:           54 FMSBCD  DS   1          SUBCODE
0000:           55 FMSBNO  EQU  $00           NO OPERATION
0001:           56 FMSBON  EQU  $01           READ/WRITE ONE BYTE
0002:           57 FMSBRA  EQU  $02           READ/WRITE RANGE OF BYTES
0003:           58 FMSBPO  EQU  $03           POSITION AND DO ONE BYTE
0004:           59 FMSBPR  EQU  $04           POSITION AND DO RANGE
0002:           60 FMPRMS  DS   8          SPECIFIC PARAMETERS

000A:           62 *            OPEN PARMS
0002:           63         ORG  FMPRMS
0002:           64 FMRCLN  DS   2          RECORD LENGTH
0004:           65 FMVOL   DS   1          VOLUME
0005:           66 FMDRV   DS   1          DRIVE
0006:           67 FMSLT   DS   1          SLOT
0007:           68 FMTYPE  DS   1          TYPE
0000:           69 FMTYPT  EQU  0             TEXT
0001:           70 FMTYPI  EQU  1             INTEGER
0002:           71 FMTYPA  EQU  2             APPLESOFT
0004:           72 FMTYPB  EQU  4             BINARY
0008:           73 FMNAME  DS   2          ADDRESS OF FILE NAME
000A:           75 *            READ/WRITE PARMS
0002:           76         ORG  FMPRMS
0002:           77 FMRCNM  DS   2          RECORD NUMBER
0004:           78 FMOFFS  DS   2          BYTE OFFSET
0006:           79 FMRALN  DS   2          RANGE LENGTH
0008:           80 FMRAAD  DS   2          RANGE ADDRESS
0008:           81 FMDATA  EQU  FMRAAD     DATA BYTE READ/WRITTEN

000A:           83 *            RENAME PARMS
0002:           84         ORG  FMPRMS
0002:           85 FMNNAM  DS   2          ADDRESS OF NEW NAME

0004:           87 *            INIT PARMS
0002:           88         ORG  FMPRMS
0001:           89 FMPAGE  EQU  FMSBCD     FIRST PAGE OF DOS IMAGE
```

```
0002:          91 *         COMMON PARMS
000A:          92           ORG   FMPRMS+8
000A:          93 FMRC      DS    1          RETURN CODE
0000:          94 FMRCOK    EQU   0             NO ERRORS
0002:          95 FMRCBO    EQU   2             BAD OPCODE
0003:          96 FMRCBS    EQU   3             BAD SUBCODE
0004:          97 FMRCWP    EQU   4             WRITE PROTECTED
0005:          98 FMRCED    EQU   5             END OF DATA
0006:          99 FMRCNF    EQU   6             FILE NOT FOUND
0007:         100 FMRCBV    EQU   7             BAD VOLUME
0008:         101 FMRCIO    EQU   8             I/O ERROR
0009:         102 FMRCDF    EQU   9             DISK FULL
000A:         103 FMRCLK    EQU   10            FILE LOCKED
000B:         104           DS    1          NOT USED
000C:         105 FMFMWA    DS    2          FILE MANAGER WORKAREA PTR
000E:         106 FMTSL     DS    2          T/S LIST PTR
0010:         107 FMBUFF    DS    2          DATA BUFFER PTR
0800:         108           DEND

0800:         110 *         LOCATE FM PARMLIST

0800:20 DC 03 112 COPY      JSR   LOCFPL     FIND PARMLIST
0803:84 00    113           STY   PTR        SET UP POINTER TO IT
0805:85 01    114           STA   PTR+1

0807:         116 *         OPEN INPUT FILE

0807:A0 08    118           LDY   #FMNAME    STORE INPUT FILE NAME
0809:A9 AC    119           LDA   #>INAME    PTR IN LIST
080B:91 00    120           STA   (PTR),Y
080D:C8       121           INY
080E:A9 09    122           LDA   #<INAME
0810:91 00    123           STA   (PTR),Y
0812:A0 07    124           LDY   #FMTYPE    BINARY FILE AS INPUT
0814:A9 04    125           LDA   #FMTYPB
0816:91 00    126           STA   (PTR),Y
0818:A2 01    127           LDX   #1         OLD FILE EXPECTED
081A:20 D3 08 128           JSR   OPEN       AND OPEN THE FILE
081D:90 03    129           BCC   INOP
081F:4C BC 08 130           JMP   ERROR      ANY ERROR IS FATAL

0822:A5 02    132 INOP      LDA   BUFP
0824:8D EA 09 133           STA   IBUFF      SAVE OPEN FILE BUFFER
0827:A5 03    134           LDA   BUFP+1
0829:8D EB 09 135           STA   IBUFF+1
082C:20 5A 09 136           JSR   REWIND     POSITION TO START OF FILE

082F:         138 *         OPEN OUTPUT FILE

082F:A0 08    140           LDY   #FMNAME    STORE OUTPUT FILE NAME
0831:A9 CA    141           LDA   #>ONAME    PTR IN LIST
0833:91 00    142           STA   (PTR),Y
0835:C8       143           INY
0836:A9 09    144           LDA   #<ONAME
0838:91 00    145           STA   (PTR),Y
083A:A0 07    146           LDY   #FMTYPE    TEXT FILE AS OUTPUT
083C:A9 00    147           LDA   #FMTYPT
083E:91 00    148           STA   (PTR),Y
0840:A2 00    149           LDX   #0         NEW FILE IS OK
0842:20 D3 08 150           JSR   OPEN
0845:90 0B    151           BCC   OUTOP
0847:A0 0A    152           LDY   #FMRC
0849:B1 00    153           LDA   (PTR),Y
084B:C9 06    154           CMP   #FMRCNF    FILE NOT FOUND?
084D:F0 03    155           BEQ   OUTOP      YES, WAS ALLOCATED THEN
084F:4C BC 08 156           JMP   ERROR
```

```
0852:A5 02      158 OUTOP   LDA  BUFP        SAVE OPEN OUTPUT FILE BUFFER
0854:8D E8 09   159         STA  OBUFF
0857:A5 03      160         LDA  BUFP+1
0859:8D E9 09   161         STA  OBUFF+1
085C:20 5A 09   162         JSR  REWIND      POSITION TO START OF FILE

085F:           164 *       READ ADDRESS/LENGTH FROM BINARY FILE

085F:A9 04      166         LDA  #4          READ 4 BYTES FIRST
0861:A0 06      167         LDY  #FMRALN
0863:91 00      168         STA  (PTR),Y
0865:A9 00      169         LDA  #0
0867:C8         170         INY
0868:91 00      171         STA  (PTR),Y
086A:20 74 09   172         JSR  READ

086D:           174 *       READ ENTIRE BINARY FILE INTO MEMORY AT $1000

086D:AD 02 10   176         LDA  BUFFER+2    COPY DATA LENGTH TO LIST
0870:A0 06      177         LDY  #FMRALN
0872:91 00      178         STA  (PTR),Y
0874:AD 03 10   179         LDA  BUFFER+3
0877:C8         180         INY
0878:91 00      181         STA  (PTR),Y
087A:18         182         CLC
087B:AD 02 10   183         LDA  BUFFER+2    COMPUTE ENDING BYTE
087E:48         184         PHA
087F:69 00      185         ADC  #>BUFFER
0881:85 04      186         STA  EBYTE
0883:AD 03 10   187         LDA  BUFFER+3
0886:48         188         PHA
0887:69 10      189         ADC  #<BUFFER
0889:85 05      190         STA  EBYTE+1
088B:20 74 09   191         JSR  READ        READ BLOB INTO MEMORY

088E:           193 *       WRITE ENTIRE BLOB OUT INTO TEXT FILE

088E:A0 00      195         LDY  #0
0890:98         196         TYA
0891:91 04      197         STA  (EBYTE),Y MARK END OF FILE
0893:68         198         PLA
0894:A0 07      199         LDY  #FMRALN+1 SET RANGE LENGTH
0896:91 00      200         STA  (PTR),Y
0898:88         201         DEY
0899:68         202         PLA
089A:91 00      203         STA  (PTR),Y
089C:20 82 09   204         JSR  WRITE       WRITE BLOB FROM MEMORY

089F:           206 *       WHEN FINISHED, CLOSE FILES

089F:AD E8 09   208 EXIT    LDA  OBUFF
08A2:85 02      209         STA  BUFP
08A4:AD E9 09   210         LDA  OBUFF+1
08A7:85 03      211         STA  BUFP+1
08A9:20 46 09   212         JSR  CLOSE       CLOSE OUTPUT FILE
08AC:AD EA 09   213         LDA  IBUFF
08AF:85 02      214         STA  BUFP
08B1:AD EB 09   215         LDA  IBUFF+1
08B4:85 03      216         STA  BUFP+1
08B6:20 46 09   217         JSR  CLOSE       CLOSE INPUT FILE
08B9:4C D0 03   218         JMP  DOSWRM      BACK TO DOS

08BC:           220 *       ERROR, PRINT "ERRXX"

08BC:A0 0A      222 ERROR   LDY  #FMRC       FIND RETURN CODE
08BE:B1 00      223         LDA  (PTR),Y
08C0:48         224         PHA
08C1:A9 C5      225 ERR     LDA  #'E         PRINT "ERR"
08C3:20 ED FD   226         JSR  COUT
08C6:A9 D2      227         LDA  #'R
08C8:20 ED FD   228         JSR  COUT
```

```
08CB:20 ED FD   229         JSR  COUT
08CE:68         230         PLA
08CF:20 DA FD   231         JSR  PRBYTE     PRINT HEX CODE
08D2:00         232         BRK              DIE HORRIBLY

08D3:           234 *       OPEN: COMPLETE PARMLIST AND OPEN FILE

08D3:AD D2 03   236 OPEN    LDA  DOSWRM+2   FIND DOS ENTRY
08D6:85 03      237         STA  BUFP+1
08D8:A0 00      238         LDY  #0
08DA:84 02      239         STY  BUFP       POINT AT BUFFER CHAIN

08DC:           241 *       SCAN DOS BUFFERS FOR A FREE ONE

08DC:B1 02      243 GBUF0   LDA  (BUFP),Y   LOCATE NEXT DOS BUFFER
08DE:48         244         PHA
08DF:C8         245         INY
08E0:B1 02      246         LDA  (BUFP),Y
08E2:85 03      247         STA  BUFP+1
08E4:68         248         PLA
08E5:85 02      249         STA  BUFP
08E7:D0 0A      250         BNE  GBUF       GOT ONE
08E9:A5 03      251         LDA  BUFP+1
08EB:D0 06      252         BNE  GBUF       GOT ONE

08ED:A9 0C      254         LDA  #12        NO FILE BUFFERS RETURN CODE
08EF:48         255         PHA
08F0:4C C1 08   256         JMP  ERR        GO PRINT MESSAGE

08F3:A0 00      258 GBUF    LDY  #0         LOOK AT FILENAME
08F5:B1 02      259         LDA  (BUFP),Y
08F7:F0 04      260         BEQ  GOTBUF     NONE THERE, FREE BUFFER
08F9:A0 24      261         LDY  #36        IT'S NOT FREE
08FB:D0 DF      262         BNE  GBUF0      GO GET NEXT ONE

08FD:A9 01      264 GOTBUF  LDA  #1
08FF:91 02      265         STA  (BUFP),Y   MARK BUFFER IN USE

0901:           267 *       FINISH COMPLETING OPEN LIST

0901:A0 00      269         LDY  #FMOCOD
0903:A9 01      270         LDA  #FMOCOP
0905:91 00      271         STA  (PTR),Y    SET OPCODE TO OPEN
0907:A9 00      272         LDA  #0
0909:A0 02      273         LDY  #FMRCLN
090B:91 00      274         STA  (PTR),Y    SET RECORD LENGTH TO 0
090D:C8         275         INY
090E:91 00      276         STA  (PTR),Y
0910:A0 04      277         LDY  #FMVOL
0912:91 00      278         STA  (PTR),Y    AND VOLUME (ANY VOL)

0914:20 E3 03   280         JSR  LOCRPL     FIND RWTS PARMS
0917:84 3C      281         STY  A1L
0919:85 3D      282         STA  A1L+1
091B:A0 01      283         LDY  #1
091D:B1 3C      284         LDA  (A1L),Y    GET SLOT*16
091F:4A         285         LSR  A
0920:4A         286         LSR  A
0921:4A         287         LSR  A
0922:4A         288         LSR  A          SLOT=SLOT/16
0923:A0 06      289         LDY  #FMSLT
0925:91 00      290         STA  (PTR),Y    STORE IN LIST
0927:A0 02      291         LDY  #2
0929:B1 3C      292         LDA  (A1L),Y    GET DRIVE
092B:A0 05      293         LDY  #FMDRV
092D:91 00      294         STA  (PTR),Y    AND SLOT

092F:           296 *       COMMON INTERFACE TO FILE MANAGER
```

```
092F:A0 1E      298 CALLFM  LDY  #30
0931:B1 02      299 CFMLP1  LDA  (BUFP),Y   GET THREE BUFFER PTRS
0933:48         300         PHA
0934:C8         301         INY
0935:C0 24      302         CPY  #36
0937:90 F8      303         BCC  CFMLP1

0939:A0 11      305         LDY  #FMBUFF+1
093B:68         306 CFMLP2  PLA
093C:91 00      307         STA  (PTR),Y    COPY THEM TO FM LIST
093E:88         308         DEY
093F:C0 0C      309         CPY  #FMFMWA
0941:B0 F8      310         BCS  CFMLP2

0943:4C D6 03   312         JMP  FM         EXIT THRU FILE MANAGER

0946:           314 *       CLOSE: CLOSE DOS FILE

0946:A0 00      316 CLOSE   LDY  #FMOCOD
0948:A9 02      317         LDA  #FMOCCL
094A:91 00      318         STA  (PTR),Y
094C:20 2F 09   319         JSR  CALLFM     CLOSE FILE
094F:90 03      320         BCC  CLOK
0951:4C BC 08   321         JMP  ERROR
0954:A0 00      322 CLOK    LDY  #0         FREE BUFFER
0956:98         323         TYA
0957:91 02      324         STA  (BUFP),Y
0959:60         325         RTS

095A:           327 *       REWIND: POSITION TO START OF FILE

095A:A0 02      329 REWIND  LDY  #FMRCNM
095C:A9 00      330         LDA  #0
095E:91 00      331 REWLP   STA  (PTR),Y    ZERO RECORD NUMBER AND..
0960:C8         332         INY
0961:C0 06      333         CPY  #FMOFFS+2 BYTE OFFSET.
0963:90 F9      334         BCC  REWLP
0965:A0 00      335         LDY  #FMOCOD
0967:A9 0A      336         LDA  #FMOCPO    POSITION OPCODE
0969:91 00      337         STA  (PTR),Y
096B:20 2F 09   338         JSR  CALLFM     EXIT VIA FILE MANAGER
096E:90 03      339         BCC  REWRTS     CHECK FOR ERRORS
0970:4C BC 08   340         JMP  ERROR
0973:60         341 REWRTS  RTS
0974:           343 *       READ: READ A RANGE OF BYTES TO $1000

0974:AD EA 09   345 READ    LDA  IBUFF      FIND PROPER BUFFER
0977:85 02      346         STA  BUFP
0979:AD EB 09   347         LDA  IBUFF+1
097C:85 03      348         STA  BUFP+1
097E:A9 03      349         LDA  #FMOCRD    READ OPCODE
0980:D0 0C      350         BNE  DOIO       GO DO COMMON CODE

0982:           352 *       WRITE: WRITE A RANGE OF BYTES FROM $1000

0982:AD E8 09   354 WRITE   LDA  OBUFF      FIND PROPER BUFFER
0985:85 02      355         STA  BUFP
0987:AD E9 09   356         LDA  OBUFF+1
098A:85 03      357         STA  BUFP+1
098C:A9 04      358         LDA  #FMOCWR    WRITE OPCODE
098E:           359 *       BNE     DOIO

098E:           361 *       DOIO: READ/WRITE A RANGE OF BYTES

098E:AC 00      363 DOIO    LDY  #FMOCOD
0990:91 00      364         STA  (PTR),Y    SET OPCODE
0992:A0 01      365         LDY  #FMSBCD
0994:A9 02      366         LDA  #FMSBRA
```

```
0996:91 00      367          STA   (PTR),Y    DO RANGE OF BYTES
0998:A0 08      368          LDY   #FMRAAD
099A:A9 00      369          LDA   #>BUFFER
099C:91 00      370          STA   (PTR),Y    RANGE ADDRESS=$1000
099E:C8         371          INY
099F:A9 10      372          LDA   # BUFFER
09A1:91 00      373          STA   (PTR),Y
09A3:20 2F 09   374          JSR   CALLFM     CALL FM TO DO I/O OPERATION
09A6:90 03      375          BCC   DOIORT
09A8:4C BC 08   376          JMP   ERROR
09AB:60         377 DOIORT   RTS

09AC:           379 *        DATA

09AC:C9 CE D0   381 INAME    ASC   'INPUT
09AF:D5 D4 A0
09B2:A0 A0 A0
09B5:A0 A0 A0
09B8:A0 A0 A0
09BB:A0 A0 A0
09BE:A0 A0 A0
09C1:A0 A0 A0
09C4:A0 A0 A0
09C7:A0 A0 A0
09CA:CF D5 D4   382 ONAME    ASC   'OUTPUT
09CD:D0 D5 D4
09D0:A0 A0 A0
09D3:A0 A0 A0
09D6:A0 A0 A0
09D9:A0 A0 A0
09DC:A0 A0 A0
09DF:A0 A0 A0
09E2:A0 A0 A0
09E5:A0 A0 A0

09E8:           384 OBUFF    DS    2
09EA:           385 IBUFF    DS    2

*** SUCCESSFUL ASSEMBLY: NO ERRORS
```

附錄B 保護磁碟的策略

由於APPLEⅡ軟體質與量的增進，非法複製這有版權的軟體便可能發生。一般軟體供應者使用了許多保護軟體的方法以防盜取。因大多數保護方案含於已修改過或顧客化的磁碟作業系統，故應對磁碟的保護措施作廣泛討論。

典型保護策略的目的在阻止未授權地複製磁碟內容，或將以BASIC所寫的軟體列印（LISTING）出來檢視。許多不同的方法早已試過，所有的方法不外乎以非標準格式（FORMAT）讀寫磁碟。若讀者對正常磁碟的格式化不清楚，可以參考第三章。

早期的做法只是單純地比較所正做的事情。每當保護方法有所改進，就有人找出破解方法。如此循環下去，較新而巧妙的策略一發展出來，短時間內就會被破解掉，之後又促使軟體供應者再想出更具技巧的系統。

在現階段可以說沒有一種磁碟保護方法無法被破解。因為磁碟必定是"可BOOT"的，意即它至少須有一段落（第零軌，第零段）能被磁碟控制卡上PROM的程式讀出。藉此可以分解磁碟中正常的段落或段落群來追蹤並探索出BOOT過程，因此這些段落便不可能受到保護了。因PROM（256個拜）的空間不足，軟體無法完全檢查每個位址欄及資料欄。利用這項優點所做的保護策略其能

力仍有限，只能包括下列特定的修改。

大多數受保護的磁碟內爲APPLE DOS的修改版。修改APPLE DOS比自己重寫要容易得多，其主要範圍不外下列所述。

現在已有的保護措施都具有一個通性，就是其磁碟中都具有一些無法由正常磁碟作業系統讀取的部分。其中位址欄及資料欄兩個邏輯區域可以加以改變。每欄具有許多拜，更改它會使段落無法讀出。(ADDRESS FIELD & DATA FIELD)

位址欄的前三拜爲$D5 / $AA/$96 。更改任一拜使DOS無法定位此位址欄而造成錯誤。將此三個拜以不同方法變化時，最須注意的是必須確定此三拜組成之整體數值在磁碟數值中的唯一性。APPLE DOS保留$D5及$AA兩拜，不用它們於儲存磁碟資料。被選用的三個拜序列除了在位址欄前面外，不得出現於軌道的任何其他地方。緊接此三個拜之後是位址資訊本身（册，軌，段，及偵錯數）。其他技巧包括更改這些資訊的順序，段落號數加倍，或以某常數變更偵錯數。任一方法都會令DOS造成I / O 的錯誤信號。位址欄最後兩個拜（ $DE/$AA ），除了值不大同外它們與開始的拜很相似。這兩個拜並不須具唯一性，因爲DOS 只讀入資訊欄後面的兩個拜以確定結束，它們並不作爲欄的定位用。

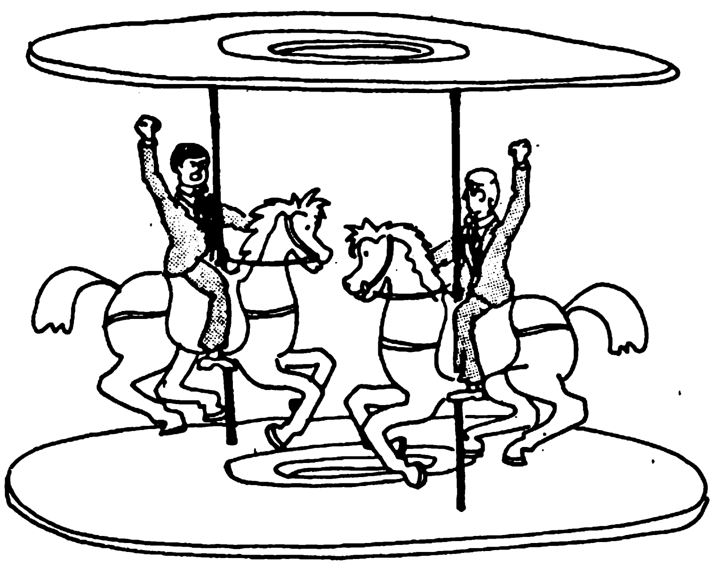
資料欄與位址欄的三個對應部分很相似。保護策略也差不多。資料欄的前三個拜$D5 / $AA/ $AD。只第三個拜與位址欄的不同，且所有用於位址欄的方法都可用於資料欄。將此二欄的第三拜交換亦爲一保護措施。資料欄的資料部分含有324個拜，其後跟著偵錯數。當寫入資料使偵錯數計算有非零結果，便發生錯誤信號。資料欄結束的拜與位址欄的同爲$DE/$AA。

前面曾提到，磁碟控制卡上的PROM忽略了這兩種欄中的特定部分。特別是既不讀入及證實結束拜（ $DE/$AA ）又不檢查

偵錯數，如此便可更改這些拜（甚至第0軌道第0段落的這些拜）。但這方法並不可靠，因爲只要對DOS的RWTS常式稍微修正就可破解它。

早期的磁碟保護措施只做簡單的修改以防複製磁碟。到現在，由於有了更多具豐富學識的使用者及更具能力的工具，常須用到多重的保護策略。一個主要方法是將控制符號植入檔案名稱中。另外，亦常見將磁碟以多重非標準格式化，甚至寫於兩軌道之間的空間上。

具技巧性的保護策略含兩個元素。其一是將資料以許多非標準化方式存入磁碟令它十分不易於被複製。其二是當重置時（RESET）更改掉部分使用到的記憶體內容（如：主要的主文頁（TEXT PAGE）或第零頁特定位置），以防止軟體被完整地自記憶體移出。



保護者與破解者之間的競賽

最近推出了許多“NIBBLE”（兩個NIBBLE爲一拜）或拜複製程式。它們與傳統上須要預知資料格式的複製程式不同，而是只要對資料結構做極少的假設。“何不將讀入記憶體的磁碟軌道內容以完全相同的方式寫至另一磁碟？”這問題自有磁碟以來便提出了。問題在於自同步（SELF-SYNC）或自動同步（AUTO-SYNC）拜上（參考第三章的討論）。這些拜含有多出的零比次（BITS）。這些零比次在資料讀入記憶體時便消失掉。而且，不可能判別出在記憶體中的16 進位制$FF是資料亦或是自同步拜。有兩種方法使用於NIBBLE 複製程式，第一種方法是藉演繹法（DEDUCTION）定出同步間隙（GAPS　）位置以找到軌道中的資料。這種方法使用在每一軌道具13 或16 個段落的磁碟上成功的機率很高（即使此磁碟修改過），但用在每段落超過256 拜的非標準段落上此法可能失效。一般而言，可以說這個方法有效但非萬無一失。第二種方法較簡單，但也有它的困難。所有軌道中的$FF 不論是否爲同步拜它都一律將之視爲資料而寫入磁碟。然而，因爲同步拜的加入須多佔25 %位置　，新磁碟要有較大的實際空間以容納軌道內容的重新寫出。假若資料中有過多$FF，軌道便會被重覆寫入。或假若磁碟推動器（DRIVER）明顯地慢於正常速度，重覆寫入的情形也會發生。於是，我們仍須回頭分析資料並做些假設。很明顯的，除了用硬體元件輔助以找尋同步拜之外，軟體程式對於資料在磁碟上的結構是必須要做些假設的。

NIBBLE 複製程式出現後又對軟體供應者造成必須要再“動手”的壓力。必須要發展出更新的方法以使NIBBLE 複製程式失效。越來越多新的保護策略加入了與“硬體”及“時序”的相關性，使得NIBBLE 複製程式必須極度地依賴使用者提供指引。如果這個趨勢繼續下去，保護策略很可能發展到以自動技術無法破解它的境界。

附錄C 名詞解釋

ACCESS TIME 存取時間	直接存取的記憶裝置（例如磁碟機）所須要用來定位及讀／寫資料的時間。
ADDRESS 位址	記憶體中資料片段所在的位置。爲自$0000至$FFFF（十進位的65535）的16進位制數目。磁碟位址（DISK ADDRESS）就是以軌道號數及段落號數表示的資料段落所在位置。
ALGORITHM 演算法	由程式或其他處理器執行並得到某些結果的一系列步驟。
ALPHANUMERIC 文數字	爲一英文字母（A－Z）或阿拉伯數碼（0－9）。此術語常用於對所有字母及數碼的通稱。
ANALOG 類比	非數位式的。如電壓或電阻具有連續的數值。
AND 及	判定兩個比次皆爲1的邏輯過程。0及1結果爲0（FALSE，假）1及1結果爲1（TRUE，眞）。
ARM	磁碟推動器的一部分，用來懸掛讀／寫頭

探臂	於磁碟表面。探臂可徑向移動以到達不同的軌道上。
ASC II	American Standard Code for Information Interchange（美國資訊交換標準碼）。以16進位制數目做爲轉換碼的安排。使一個拜可能有的256個值各代表一個字母，阿拉伯數碼，特殊字，或控制符號。ASC II 用於與週邊機的界面，例如鍵盤，印表機，或視像顯示器。
ASSEMBLY LANGUAGE 組合語言	亦稱爲機器語言（MACHINE LANGUAGE）。是計算機最原始的語言。組合語言源於機器而不同於BASIC，PASCAL，或FORTRAN 等源於爲人類的使用而設計者。組合語言編譯器（ASSEMBLER）將組合語言的敍述轉爲可由機器執行的機器碼程式。
BACKUP 備用	將程式或資料複製以防意外遺失或遭破壞的過程。
BASE 底數	使用的數目系統。10進位制（DECIMAL）是以10爲底數的數目系統。其每位數碼（DIGIT）代表一個10的級數（1，10，100，……）16進位制（HEXADECIMAL）是以16爲底數的數目系統（1，16，256，……）。二進位制（BINARY）是以2爲底數的數目系統（1，2，4，8，……）。

BINARY 二進位制	以2的級數爲底的數目系統。其數碼爲0與1。例如101是1個1，0個2，及1個4。就是10進位數的5。
BIT 比次	一個二進位制數碼(1或0)。比次是計算機的最小儲存單位及資訊單位。
BIT CELL 比次胞	磁碟上兩個時序脈衝之間用來儲存容納一個比次的空間。
BIT SLIP MARKS 比次滑動記號	磁碟欄的後註(EPILOGUE)，用於覆核磁頭是否仍處於同步讀入狀態及確定段落未受傷害。
BOOT/BOOTSTRAP 靴式激發	以連續地載入漸大程式片段的方法，將磁碟中極大的程式完全載入記憶體的過程。每個已載入的程式負責載入下一個更大的程式。也就是一種"將自己以靴帶式漸次自磁碟拉出"的程式載入法。
BRK 間斷	BRACK。爲一組合語言指令，用於強迫中斷(INTERRUPT)及立即凍結程式之執行。
BUFFER 緩衝器	記憶體的某位置用來暫存來自或傳送至週邊裝置的資料(如磁碟)。
BUG 錯誤	程式的錯誤或程式的錯誤運算。
BYTE 拜	計算機中可定位址的最小記憶體單位。一個拜常包含8個比次，能表示出10進位制的0-255數目或一個文數字。

CARRIAGE RETURN 回車(C/R)	命令印表機結束一行並自下一行開始打的控制符號。在列印時，回車之後常隨一個換行(LINE FEED)。
CARRY FLAG 進位旗號	6502的旗號之一，顯示加法是否造成進位。也常用爲作系統程式的錯誤指示器。
CHAIN 鏈	資料元素(ELEMENTS)的鏈結表。儲存體不連續的資料元素經由存在前一資料元素的後一資料元素位址指標而相連接。
CHECKSUM/CRC 偵錯數 cyclic redundency check	用於查證資料是否傳送良好。寫入資料時，所有拜的加法總和亦隨著儲存起來(稱爲偵錯數)。若讀出時所得之和與偵錯數不符合則可得知資料已遭到破壞。
CLOBBERED 傷害	受傷害或被破壞。受傷害的段落就是段落遭重覆寫入，使原有的內容無法還原。
CODE 碼	計算機可執行的指令碼。常就是機器語言碼。
COLDSTART 冷啓動	爲一種重新啓動程式。執行時將所有的參數重新起始設定，並放棄正進行的程式。
CONTIGUOUS 緊接的	實質上相鄰接的。在記憶體或磁碟中兩個相鄰接的拜，稱爲CONTIGUOUS。
CONTROL BLOCK 控制塊	作業系統處理資源(RESOURCES)時所使用的一群資料集。例如檔案緩衝器。
CONTROL CHARACTER控制符號	使週邊機執行一簡單動作的特殊ASCII碼，此碼無法被印出。
CONTROLLER CARD 控制器卡	插於APPLE連接器(CONECTOR)用來與週邊裝置(如磁碟機，印表機等)通

	信的硬體電路板，常具有推動程式於其ROM中。
CSWL	零頁中的一向量指標，輸出的資料經由它被傳送到CRT顯示器或印表機。
CYCLE 週	計算機中央處理器的最小時間單位。做完每個機器語言指令須用兩至三週。APPLE的一週為1微秒，即百萬分之一秒。
DATA 資料	資訊的單位。
DATA SECTOR BUFFER 資料段落緩衝器	APPLE DOS使用256個拜的緩衝器以保存所須的磁碟段落映像。在檔案讀資料時，緩衝器內容漸次取出，直到取盡後緩衝器便再度存入下一個段落映像。
DATA TYPE 資料型式	拜的資訊型式，包括可印出的ASCⅡ符號，二進位制資料，或機器語言碼指令。
DCT 裝置特性表	為RWTS所用的輸入參數表，它表明了磁碟推動器的硬體特性。
DECIMAL 10進位	以10的級數為底的數目系統。數碼為0到9。
DEFERED COMMANDS延遲命令	執行BASIC程式時常用到的DOS命令。如OPEN，READ，WRITE，及CLOSE都是延遲命令。
DIGITAL 數值	非類比式的。為不連續的離散數值。僅數位數值才能存於計算機中。實際的類比信號如電壓大小，光度等必先轉換成一數目並以“捨入”法使其成為離散數值，然後

	才能輸入計算機。
DIRECT ACCESS 直接存取	能快速存取任何位置之資料片段的週邊儲存裝置。磁帶非直接存取儲存裝置，因整條磁帶須全部讀完才能夠取得最後一個拜的資料。至於磁碟則是直接存取儲存裝置，因其探臂可迅速移動至任意軌道及段落上取得任意位置的資料。
DIRECTORY 索引表	磁碟中全部檔案的目錄。索引表必須含檔案名稱，檔案在磁碟中的位置，及關於儲存資料型式的相關資訊。
DISK INITIALIZ-ATION 起始設定磁碟	將軌道格式（FORMAT）資訊（包括段落及間隙（GAPS）放入空白磁碟的過程。當起始設定磁碟時，DOS 亦將 VTOC 及索引表放進新的格式化磁碟並保存HE-LLO 程式於其上。
DISPLACEMENT 位移量	特殊拜或欄與所在的資料節（BLOCK）起始位置之間距離。位移量0代表第一個拜，位移量1代表第二個拜，……等等。位移量亦稱偏移量（OFFSET）。
DRIVER 推動器	提供資料串給其他程式或輸出裝置的程式。印表機推動器將來自使用者程式的資料排成線列，然後送到印表機印之。
DUMP 傾印	將記憶體或磁碟內容以 16 進位制未格式化或半格式化列表印出。常用於診斷偵錯。
ENCODE 編碼	依須要將資料自一型式轉換爲另一型式。DOS3.3 的資料，由原 8 比次組成的拜轉

	成 6 比次組成的拜以存入磁碟中。
ENTRY POINT (EPA) 進入點	執行程式的起始位置。並不一定同於載入點(程式在記憶體中最低位置)。
EOF 檔案結束	END OF FILE。這標誌表示資料檔案已結束。APPLE DOS 的主文檔案(TEXT FILE)中此標誌為 $00 。
EPILOGUE 後註	軌道上某欄的最後三個拜。這些拜確定了在它們之前的資料的完整性。
EXCLUSIVE OR 互斥或	判明兩比次不同值的邏輯運算。1 EOR 0 得 1 ,1 EOR 1 得 0 。
FIELD 欄	一群連續的拜所構成的單一資料片段。例如人的名字,年齡,或其人的社會保險號碼等。格式化磁碟時,一群資料拜夾於兩間隙之間。
FILE 檔案	磁碟,或其他儲存大量資料的媒體之中被賦予名稱的資料集。檔案內容可為資料或為程式。
FILE BUFFER 檔案緩衝器	APPLE DOS 處理開檔案(OPEN FILE)所用的一群緩衝器。包括一資料段落緩衝器,一軌道/段落表之段落緩衝器,一檔案處理器工作區緩衝器,檔案名稱及指標(POINTERS)。DOS 的 MAXFILE 3 命令造成 3 個可配置(ALLOCATE)的檔案緩衝器。
FILE DESCRIPTOR 檔案說明	描述一檔案的磁碟檔案索引表的一進入點。包括檔案名稱,資料型式,檔案長度及

	其在磁碟中之位置。
FILE MANAGER 檔案處理器	DOS處理檔案的部分。檔案處理器處理類如OPEN，CLOSE，READ，WRITE，POSITION，RENAME，DELETE等功能。
FILE TYPE 檔案型式	檔案的資料型式。BINARY，APPLESOFT，INTEGER BASIC，TEXT，RELOCATABLE，S，A，及B型式都是用於APPLE DOS的檔案型式。
FIRMWARE 堅體	介於軟體與硬體之間。是指存於ROM中的"微碼"或"微程式"。
GAPS 間隙	磁碟資料欄之間的空間。APPLE磁碟的間隙含有自同步拜。
HARD ERROR 硬性錯誤	無法復原的輸出入錯誤。存在磁碟段落中之資料無法再度順利讀出。
HARDWARE 硬體	計算機的實體裝置。不包括在此裝置上使用的程式。例如磁碟機就是一硬體元件。
HEAD 磁頭	磁碟機的讀/寫頭。磁性拾器（像立體音響盤式磁帶機上的磁頭）。它停在旋轉的磁碟表面。
HEXADECIMAL/HEX 16進位	以16的級數爲底的數目系統。數碼有0到9及A到F，其中A表10，B表11，……，F表15。B30就是11個256，3個16及0個1，即10進位制的2864。兩個16進位制數碼合起來表示一個拜的內容。16進位制因易於與二進位制互

	相轉換，故用於計算機。
HIGH MEMORY 高位址記憶體	具有高位址值的記憶體位置。$FFFF 是最高記憶體位置，即記憶體之頂端。
HIMEM	APPLE DOS第零頁的位址，即爲可用來存BASIC 程式及其變數的最高記憶體之次一個更高位置拜的位址。
IMMEDIATE COMMAND 立即命令	可於任意時間鍵入並執行的DOS命令，特別是發生在DOS等待鍵盤命令時。與立即命令相對應的爲延遲命令（DEFERED COMMAND）
INDEX 索引	一固定位置量，指入表格（TABLE）或儲存節（STORAGE BLOCK）之中。
INSTRUCTION 指令	執行組合語言程式或機器語言程式時的一單純步驟。指令有加，減，存（STORE），載（LOAD）等功能。
INTEGER 整數	相對於浮點數（FLOATING POINT），是個無分數的"整個"數目。
INTERCEPT 截斷器	安插於別的程式中間執行的程式。例如視像截斷器（VIDIO INTERCEPT）用來重覆引導自顯示幕至印表機的程式輸出。
INTERLEAVE 交錯	磁碟軌道上各段落的實際安排順序是互相交疊的，以消除旋轉延遲，將所須存取時間降至最低。亦稱偏置（SKEWING）或交織（INTERLACING）。
INTERRUPT 中斷	令計算機暫停執行程式並進入特殊處理常式的硬體信號。用於即時處理的時段結束

	之時(REAL-TIME CLOCK TIME-OUT),BRK指令,及重置(RESET)等工作。
IOB 輸出入節	INPUT/OUTPUT BLOCK。傳送給RWTS使用的參數資料集,用以描述即將執行的動作。
I/O ERROR 輸出入錯誤	INPUT/OUTPUT ERROR。與週邊裝置(如磁碟)互傳資料時發生的錯誤。
JMP 跳越	6502的一組合語言指令,使程式跳至不同記憶位置去執行。
K	數量單位。1 K拜即1024個拜。
KSWL	來自鍵盤或遠方終端機的資料可經由這個零頁中的向量指標而順利通過。
LABEL 標誌	程式或記憶體中某位置的相關名稱。類似BASIC中的敍述數(STATEMENT NUMBER)一樣,標誌使用於組合語言程式中。
LATCH 閂	儲存一個拜的輸出入硬體元件。此拜保留在閂中直到中央處理器來讀它爲止,反方向亦然。
LINK 鏈結	資料或緩衝器鏈(CHAIN)上每個元素單位(資料或緩衝器單位)之中存的一位址指標,此指標指向下一個元素的起始位置。
LIST 表	一次元資料項目的列陣。

LOAD POINT(LP) 載入點	組合語言程式載入記憶體的最低位址(第一拜位址)。它不一定同於進入點(EP)位址。
LOGICAL 邏輯的	只對二進位制"眞"或"假"1或0運算的演算法。AND,OR,NAND,NOR,及EOR都是邏輯的運算。
LOOP 環路	包含一群指令或敍述並爲重覆執行的程式結構。
LOW MEMORY 低位址記憶體	具低位址的記憶位置。$0000是最低記憶位置,稱爲記憶體底部。
LOMEM	可用來存BASIC 程式及其變數的第一個記憶體拜位址。
LSB/LO ORDER	最低位比次或最低位拜(LEAST SIGNIFICANT BIT或LEAST SIGNIFICANT BYTE)。拜中代表"1"的比次,或16進位制位址的第二個拜,$8030中$30爲低位拜。
MASTER DISK 主磁碟	可在各種APPLE系統BOOT,並可發揮完全功能的DOS磁碟。
MICROSECOND 微秒	百萬分之一秒。爲APPLE中央處理器之一週時序。記爲"μSec"。
MONITOR 監督程式	已建在計算機中的程式,開機時先取得中央處理器的控制權。APPLE及監督程式存於ROM中,可用來檢查及修改記憶體之拜。
MSB/HI ORDER	最低位比次或最低位拜(MOST SIGNIFICANT BIT或MOST SIGNIFI-

	CANT BYTE)。拜中代表"128"的比次，或16進位制位址的第一個拜，拜$83中MSB爲1。
NULL 零的	空的，沒有長度或數值的。零串(NULL STRING)不含任何文數字(CHARAC-TER)。零控制符號($00)(NULL CONTROL CHARACTER)對印表機無作用，亦即使其空閒著(IDLE)。
NIBBLE/NYBBLE	拜的一部分，含4比次以表一個16進位制數碼。$FE含兩個NIBBLE，$F及$E。
OBJECT CODE 目標碼	具二進位制數碼，可執行的機器語言程式，爲組合語言編譯器的輸出。
OBJECT MODULE 目標模組	具目標碼型式的完整機器語言程式，爲磁碟的一檔案。
OFFSET 偏移量	特殊拜或欄與所在的資料節(BLOCK)起始位置之間距離。偏移量0表第1拜，偏移量1表第2拜，等等。亦稱爲位移量(DISPLACEMENT)。
OPCODE 運算碼	OPERATION CODE。三個字母組成的記號代表一個組合語言指令。JMP是跳越指令的運算碼。
OPERATING SYSTEN 作業系統	自動處理記憶體及週邊裝置的機器語言程式，用來減輕應用程式師的負擔。
OR 或	比較是否兩個比次之一或兩者皆爲1的邏輯運算。1 OR 1得1（眞），1 OR 0

	得1，0 OR 0得0（假）。
OVERHEAD	系統處理記憶體或磁碟所須的記憶體或磁碟的空間。磁碟索引表（DIRECTORY）及VTOC都是磁碟的OVERHEAD。
PAGE 頁	記憶體中具有相同高位拜位址的256個拜。零頁即記憶體的第一個256拜（$0000到$00FF）。
PARALLEL 平行的	相對於"串列的"。同時將拜中所有比次傳送出的通信模式，每個比次使用一條獨立傳送線。
PARAMETER LIST 參數表	呼叫程式與副程式之間通信所使用的儲存地區。參數表內含副程式所用的輸入及輸出變數。
PARITY 同位	與偵錯數有相同偵錯功能的一比次，此比次的值爲拜中每一比次的和，它附在拜的後面傳送。"同位"用於較貴重的記憶體以檢出或更正一比次的錯誤，當通信傳輸時並可檢出雜訊造成的錯誤。
PARSE 剖析	翻譯一文字資料串的過程，例如一具有關鍵字的命令。
PATCH 修補	對組合語言程式的目標碼作小量更改。亦稱"ZAP"。
PERIPHERAL 週邊機	計算機外部的裝置，如磁碟或印表機。亦稱爲輸入／輸出裝置。
PHYSICAL RECORD 實體資料錄	對應於週邊裝置最小儲存單位的一個資料集。磁碟的實體資料錄爲一段落。

POINTER 指標	一資料節或單一資料項目的位址或記憶位置。此位址指向該資料。
PROLOGUE 前序	一磁碟欄開始的三個拜，用來在軌道中分辨出一欄的位置。
PROM 可程式化僅讀記憶體	PROGRAMMABLE READ ONLY MEMORY。PROM 常用在週邊裝置的控制卡上，以保存一個處理此裝置與應用程式間界面的程式。
PROMPT 催促	通知使用者輸入資料的一串輸出文字。APPLE 監督程式的催促符號爲“ * ”。
PROTECTED DISK 受保護磁碟	一磁碟，其格式或內容已修改過以防被複製。現在大多數的零售軟體都存於受保護的磁碟以防盜竊。
PSEUDO-OPCODE 假運算碼	不轉換成機器指令的特殊組合語言運算碼。假運算碼用來指示組合語言編譯器以執行許多功能，如組合語言程式列印時跳略掉某頁，或在輸出目標碼中保留資料空間。
RANDOM ACESS 隨機存取	直接存取，具有不須依順序而迅速取得儲存體內任一資料片段的能力。
RAM 隨機存取記憶體	RAMDOM ACESS MEMORY。可利用位址以存入及取出資料值的計算機記憶體。
RECAL 重校準	重校準磁碟探臂使讀寫頭定位於第零軌道之上。將探臂向外拉出磁碟邊，直到發出一“咔啦”聲。
RECORD 資料錄	一聚集資料項目或資料欄。一個或數個資料錄常集成一檔案。比如每個資料錄可能

	對應一個職員的資料。
REGISTER 暫存器	中央處理器的一個暫時儲存位置的稱呼。6502有A，X，Y，S，及P五個暫存器。組合語言常利用暫存器以存取記憶體及作算數運算。
RELEASE 發行	軟體已散播部分的版本。DOS 已有許多發行的版本。
RELOCATABLE 可重定位	一具有機器語言程式的目標碼模組檔案(OBJECT MODULE PILE)，它能在任何記憶體位置執行。
RETURN CODE 返回碼	自副程式返回時所得的一數值，用以顯示執行的成敗。"零"返回碼常代表執行無誤，其他的返回碼之值依副程式不同各代表不同的錯誤原因。
ROM 僅讀記憶體	READ ONLY MEMORY。具永久固定值的記憶體。APPLE監督程式及BASIC翻譯器都存在ROM中。
RWTS 讀／寫，軌道／段落	軌道及段落的一副程式集。RWTS是DOS一部分並可由外界組合語言程式呼叫之。
SEARCH 搜尋	掃描軌道以尋獲一指定段落的過程。
SECTOR 段落	磁碟軌道上可更新資料的最小單位。APPLE DISKⅡ 的每一段落含256個拜。
SECTOR ADDRESS 段落位址	內含册號數，軌道號數，段落號數的一磁碟欄，用來認定一段落資料欄。

SECTOR DATA 段落資料	含有以分塊(NIBBILIZED FORM)型式的實際段落資料的一磁碟欄。
SEEK 尋找	移動磁碟探臂到指定軌道的過程。
SELF-SYNC 自同步	亦稱為"AUTO-SYNC"自動同步。多於8個比次的特殊磁碟拜，在讀入時用來使硬體與拜的邊界同步。
SEQUENTIAL ACCESS 連續存取	為一資料獲取的方式，就是讀取資料的順序與寫至磁碟時相同。
SERIAL 串列的	對應於"平行的"。經由一條線每個時間傳送一個比次的通信方式。
SHIFT 移動	將拜中的各比次全部向左或向右移動一個位置，並於另一邊的空比次放入"零"的一種邏輯運算。
SLAVE DISK 副磁碟	一個具有DOS複製品而不可重定位(RELOCATABLE)的磁碟。不論所在機器的記憶體大小如何，它將DOS 映像載入記憶體固定位置。
SOFT ERROR 軟性錯誤	可補救的輸出入錯誤。舊磁碟偶而會造成軟性錯誤。
SOFTWARE 軟體	可載入記憶體並執行的計算機程式及資料。
SOURCE CODE 本源碼	人可了解的程式型式，是文字型式而非內部二進位機器碼型式。本源組合語言碼須經組合語言翻譯器轉換成機器碼或目標碼(OBJECT CODE)。

SKEWING 偏置	將段落交錯的過程。參考"INTERLEAVE(交錯)"。
STATE MACHINE 狀態機	定義爲單一目標狀態(UNIQUE TARGET STATE)的處理器(軟體或硬體處理器),它供給了一輸入狀態及一些特定的情況。DOS 以狀態機的方式以保持視像截斷器的軌道,及使用磁碟控制卡硬體處理磁碟資料。
STROBE 探觸	短暫地查照一特殊I/O位址以觸發I/O功能的動作。亦稱"TOGGLING觸動"。STROBE$C030 會在喇叭上造成一咔啦聲。
SUBROUTINE 副程式	一程式具有主程式執行時常被重複使用的功能,主程式必須在許多地方呼叫它。
TABLE 表	具有類似的格式,存在記憶體中的一群資料進入點(DATA ENTRIES)集合。例如每個進入點包含一程式名稱及其位址。可以檢視此表以獲得已知名稱程式的位址。
TOGGLE 觸動	參照STROBE。
TOKENS 代號	以壓縮所佔用的記憶體空間,或加快處理速度的一種方法。
TRACK 軌道	磁碟上一完整的磁性儲存體環路。一個APPLE磁碟有35個同心環軌道。
TRANSLATE TABLE	以一對一取代輸入碼的單一拜碼的表格。

轉換表	轉換表用來將6比次碼轉成磁碟碼。
T／S　LIST	TRACK／SECTOR　LIST。一描敍檔案所在位置的段落，其描敍法爲，將檔案各資料段落的軌道號數及段落號數以應讀或寫的順序列表出來。
TTL	TRANSISTOR　TO　TRANSISTOR　LOGIC。積體電路之間相接的一種標準，它亦定義了表示“0”與“1”的電壓值。
UTILITY 公用程式	維持或支援其他程式或磁碟檔案之發展的程式。
VECTOR 向量指標	一群在記憶體中固定位置的指標或JMP指令集，用來取得可重定位的程式或資料
VOLUME 册	一個含有單一或許多檔案的磁碟，碟盤或卡帶的辨證法。
VTOC 册內容表	VOLUME　TABLE　OF　CONTENTS。依據IBM　OS／VS　VTOC。APPLE上，有一段落存着磁碟中自由段落（FREE　SECTORS）的映圖並提供索引表（DIRECTORY）位置。
WARMSTART 暖啓動	一程式重新啓動（RESTART），但仍盡可能保留正進行的工作。DOS　暖啓動保留了記憶體中的BASIC程式。
WRITE　PROTECTED 具有寫入保護	一磁碟的寫入保護缺口被蓋上以防磁碟推動器寫入資料。
ZAP	源於JBM公用程式SUPERZAP　。爲一

	使用 16 進位數更新磁碟中的“拜”的程式。
ZERO PAGE 第零頁	記憶體中第一個 256 拜。第零頁位置對中央處理器有特殊意義，它們的處理與安排十分重要。

内部交流

G 19/13 APPLE DOS

磁碟作业系统详论

(中3—5/77)

D00180